Mission Research Corporation

MRC-COM-R-96-0461

PHASE I FINAL REPORT

DEVELOPMENT OF AN
URBAN WARFARE VIRTUAL ENVIRONMENT

MODELING AND SIMULATION OF WEAPONS EFFECTS ON BUILDING STRUCTURAL INTEGRITY AND PERSONNEL FOR DIS VIRTUAL ENVIRONMENTS

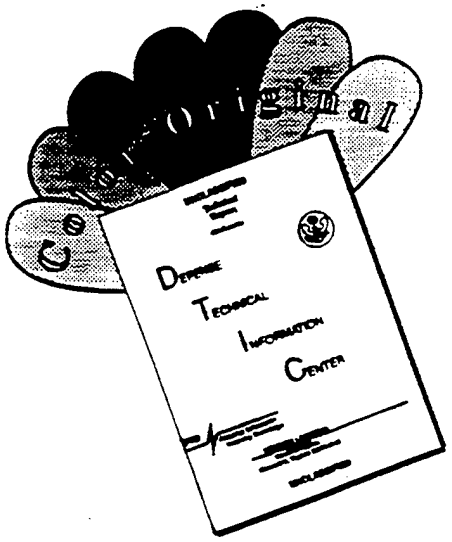
30 SEPTEMBER 1996

R. D. Eisler, A. K. Chatterjee,
D. Vaske, and G. H. Burghart
MISSION RESEARCH CORPORATION
17150 NEWHOPE STREET, SUITE 516
FOUNTAIN VALLEY, CA 92708-4253

CONTRACT M67004-96-C-0013 (CDRL 2AB)

U.S. ARMY SIMULATION, TRAINING AND
INSTRUMENTATION COMMAND (STRICOM)
ATTN.: AMSTI-ET (ADMIRAL PIPER)
12350 RESEARCH PARKWAY
ORLANDO, FL 32826-3276

DISCLAIMER NOTICE



THIS DOCUMENT IS BEST QUALITY AVAILABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF COLOR PAGES WHICH DO NOT REPRODUCE LEGIBLY ON BLACK AND WHITE MICROFICHE.

MRC-COM-R-96-0461

PHASE I FINAL REPORT

DEVELOPMENT OF AN URBAN WARFARE VIRTUAL ENVIRONMENT

MODELING AND SIMULATION OF WEAPONS EFFECTS ON BUILDING STRUCTURAL INTEGRITY AND PERSONNEL FOR DIS VIRTUAL ENVIRONMENTS

30 SEPTEMBER 1996

R. D. Eisler, A. K. Chatterjee,
D. Vaske, and G. H. Burghart
MISSION RESEARCH CORPORATION
17150 NEWHOPE STREET, SUITE 516
FOUNTAIN VALLEY, CA 92708-4253

CONTRACT M67004-96-C-0013 (CDRL 2AB)

U.S. ARMY SIMULATION, TRAINING AND INSTRUMENTATION COMMAND (STRICOM)
ATTN.: AMSTI-ET (ADMIRAL PIPER)
12350 RESEARCH PARKWAY
ORLANDO, FL 32826-3276

REPORT DOCUMENTATION PAGE

Form Approved
OMB NO. 0704.0188

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	30 Sept. 1996	**FINAL REPORT: 5 FEBRUARY 1996 – 4 AUGUST 1996**

4. TITLE AND SUBTITLE **DEVELOPMENT OF AN URBAN WARFARE VIRTUAL ENVIRONMENT: MODELING AND SIMULATION OF WEAPON EFFECTS ON BUILDING STRUCTURAL INTEGRITY AND PERSONNEL FOR DIS VIRTUAL ENVIRONMENTS**

5. FUNDING NUMBERS

6. AUTHOR(s)
R. D. EISLER, A.K. CHATTERJEE, D. VASKE, AND G.H. BURGHART

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
MISSION RESEARCH CORPORATION
17150 NEWHOPE STREET, SUITE 516
FOUNTAIN VALLEY, CA 92708-4253

8. PERFORMING ORGANIZATION REPORT NUMBER
MRC-COM-R-96-0461

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. ARMY SIMULATION, INSTRUMENTATION AND TRAINING COMMAND (STRICOM);12350 RESEARCH PARKWAY; ATTENTION: AMSTI-ET (ADMIRAL PIPER); Orlando, Florida 32826-3276

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

This effort will incorporate real-time simulation of weapons effects into a DIS/HLA compatible urban warfare virtual environment. The Phase I effort identified key problems and relevant approaches for Phase II. Moreover, fundamental modeling and simulation issues which extend beyond the current application were addressed; i.e., how to embed modeling of physical processes (which is necessary if the objects in a simulation are to interact in a non-scripted manner) in real-time visual simulations. During Phase I, a "precision" MOUT scenario for clearing a building was developed. This scenario included interior and exterior construction details, weapons and ammunition, soldier tasks and engagement ranges. Ballistic experiments were conducted on selected weapon-target combinations to identify interaction dynamics, and provide data for texture maps and analytical models. A sub-real-time code describing external and terminal ballistics of small arms was developed and correlated with experimental data. A stochastic model was developed to describe distribution of glass fragments from an incident blast wave. An approach was developed to describe structural integrity in explosive weapon effects environments using parametric solutions from hydrocodes and handbook solutions. A sub-real-time methodology using the MRC *John O.* phantom, which will be interfaced with the Phase II virtual environment, will be employed to describe weapon effects on personnel. Finally, a digital fly-through model of the McKenna Town Hall was developed and various weapon effects were statically simulated.

(REPORT DEVELOPED UNDER SBIR CONTRACT)

14. SUBJECT TERMS
MOUT **Virtual Environment** **Weapon Effects** **Casualty Analysis** **Virtual Reality**
Building **SBIR Report** **Visualization** **Urban Terrain** **Small Arms**

15. NUMBER OF PAGES
86

16. PRICE CODE

17. Security CLASSIFICATION OF REPORT	18. Security CLASSIFICATION OF THIS PAGE	19. Security CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	**UNCLASSIFIED**	**UNCLASSIFIED**	**UNLIMITED**

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. BACKGROUND AND SUMMARY OF PHASE I ACCOMPLISHMENTS

This effort will incorporate real-time simulation of weapons effects into a DIS/HLA compatible urban warfare virtual environment. The Phase I effort identified key problems and relevant approaches for Phase II. Moreover, fundamental modeling and simulation issues which extend beyond the current application were addressed; i.e., how to embed modeling of physical processes (which is necessary if the objects in a simulation are to interact in a non-scripted manner) in real-time visual simulations.

Incorporation of realistic, physically based, weapon effects into virtual environments that can be used to train soldiers and law enforcement personnel represents a *holy grail* for the military modeling and simulation community. Currently, visual representations of weapon effects are superimposed on visual simulations almost as an "after thought." As a consequence, physical processes associated with employing a weapon are not incorporated into the simulation. For example, compelling visual representations of the muzzle flash and plume effects from howitzers might be included in a war gaming virtual environment but the effects of cratering, debris, and ejecta (the purpose of firing a howitzer) is not evident on the characters or military assets in the simulation.[1] So, while photorealistic, visual animation of hypothetical weapon effects might be represented or "scripted" within a simulation, the effect on other objects or non-scripted events becomes problematic.

At least three obstacles thwart incorporation of physically based weapon effects on building structural integrity and personnel in real-time simulations. First, most visualization tools merely represent visible geometric surfaces and not mechanical or structural properties necessary to analyze structural integrity or weapon interaction. Second, whereas visualization tools effectively represent rigid-body displacement (translation and rotation) these tools are limited in representing non-rigid body deformation (e.g., crushing, fragmentation, and bending). Finally, the resolution entailed in describing weapon effects and their interaction with objects in the simulation is inconsistent with real-time performance of a high fidelity visual model.

The need to realistically simulate weapon effects with sufficient generality and degrees-of-freedom in a training environment has never been more important than in an urban warfare setting where the rules-of-engagement, limits on collateral damage to non-combatants and the urban infrastructure, and weapon performance and effects can vary widely.

Collateral effects of weapons, e.g., fragmentation and backblast, can have disastrous and unexpected consequences in these environments where target engagement

[1] Example provided by D. Reiss, U.S. Army Infantry School, personal communication, 23 April 1996.

ranges can be very close. Urban terrain also provides tremendous variation in terms of weapon effects. For example, the same weapon employed against two different wall constructions can have completely different effects – e.g., ricochet versus penetration. A 5.56-mm rifle bullet striking plate glass windows can fragment, ricochet, or be deflected during perforation as a function of striking obliquity.[2] Engagement ranges and times are short resulting in a fast and confusing pace of battle where the risk of friendly fire, ricochet, fratricide, and injury to non-combatants is high.

Addressing these issues entails a high degree of situational awareness and mission planning. This can only be realized by mission rehearsal for all the unexpected contingencies that an urban warfare environment can provide. Further, these contingencies, particularly as related to weapon effects, cannot be safely employed in a MOUT training course, rather, the only option to safely demonstrate collateral weapon effects is in a virtual training environment. Table 1 includes additional considerations relative to combat in built-up areas and employment of weapons.

A Phase I SBIR effort that addresses these issues was awarded to Mission Research Corporation (MRC) in February 1996 and concluded August 1996. Mr. Admiral Piper was the U.S. Army Simulation, Training, and Instrumentation Command's (STRICOM's) Contracting Officer's Representative. The Phase I effort included eight tasks with a six month period of performance for the technical effort.

In Task 1 a scenario was identified that consisted of clearing the Town Hall at the Fort Benning McKenna MOUT site. We investigated to what extent the McKenna Town Hall represented a typical structure and decided to provide user prescribed options to change interior and exterior construction details. We also developed a list of weapons and ammunition that might be used in a "precision" MOUT building clearing operation.

In Task 2, we conducted ballistic experiments to understand the dynamics of the interaction of various weapons with typical interior building construction. We also used the data to provide texture maps for the Task 6 visualization effort and to develop analytical models in the Task 4 Structural Integrity effort.

In Task 3 we developed a sub-real-time code which describes the external ballistics of various small arms of interest. This code was favorably correlated with experimental data and more detailed analytical approaches. We also developed an approach for describing the free-field environment of fragmenting munitions and explosives. For fragmenting munitions we use a data base of arena test data. For explosives we use modified Sach's scaling of a standard blast wave to obtain the initial pulse incident on a target.

---

[2] S.C. Robertson, "Rifle Ammunition Performance through Barriers," INTERNATIONAL WOUND BALLISTICS ASSOCIATION REVIEW, 2(4), 1996.

Table 1. Important Characteristics of Combat in Urban Areas[3]

a. Hard, smooth, flat surfaces are characteristic of urban targets. Rarely do rounds impact perpendicular to these flat surfaces but at some angle of obliquity. This reduces the effect of a round and increases the threat of ricochets. The tendency of rounds to strike glancing blows against hard surfaces means that up to 25 percent of impact-fuzed explosive rounds may not detonate when fired onto rubbled areas.

b. Engagement ranges are close. Studies and historical analyses have shown that only 5 percent of all targets are more than 100 meters away. About 90 percent of all targets are located 50 meters or less from the identifying soldier. Few personnel targets will be visible beyond 50 meters and usually occur at 35 meters or less. Minimum arming ranges and troop safety from backblast or fragmentation effects must be considered.

c. Engagement times are short. Enemy personnel present only fleeting targets. Enemy-held buildings or structures are normally covered by fire and often cannot be engaged with deliberate, well-aimed shots.

d. Depression and elevation limits for some weapons create dead space. Tall buildings form deep canyons that are often safe from indirect fires. Some weapons can fire rounds to ricochet behind cover and inflict casualties. Target engagement from oblique angles, both horizontal and vertical, demands superior marksmanship skills.

e. Smoke from burning buildings, dust from explosions, shadows from tall buildings, and the lack of light penetrating inner rooms can combine to reduce visibility and to increase a sense of isolation. Added to this is the masking of fires caused by rubble and man-made structures. Targets, even those at close range, tend to be indistinct.

f. Urban fighting often becomes confused melees with several small units attacking on converging axes. The risks from friendly fires, ricochets, and fratricide must be considered during the planning phase of operations and control measures continually adjusted to lower these risks. Soldiers and leaders must maintain a sense of situational awareness and clearly mark their progress in accordance with unit standard operating procedures to avoid fratricide.

g. Both the firer and target may be inside or outside buildings, or they may both be inside the same or separate buildings. The enclosed nature of combat in built-up areas means that the weapon's effect, such as muzzle blast and backblast, must be considered as well as the round's impact on the target.

h. Usually the man-made structure must be attacked before enemy personnel inside are attacked. Therefore, weapons and demolition can be chosen for employment based on their effects against masonry and concrete rather than against enemy personnel.

i. Modern engineering and design improvements mean that most large buildings constructed since World War II are resilient to the blast effects of bomb and artillery attack. Even though modern buildings may burn easily, they often retain their structural integrity and remain standing. Once high-rise buildings burn out, they are still useful to the military and are almost impossible to damage further. A large structure can take 24 to 48 hours to burn out and get cool enough for soldiers to enter.

j. The most common worldwide building type is the 12 - to 24 - inch brick building. Table 5, page 13, lists the frequency of building types world-wide.

[3] Source: "An Infantryman's Guide to Combat In Built Up Areas," FM 90-10-1, Department of the Army, 12 May 1993

In Task 4, a terminal ballistics code was developed and programmed for small arms of interest. Code predictions were correlated with Phase I ballistic experiments performed in Task 2. Parametric solutions were programmed for other terminal ballistics phenomenology (e.g., ricochet) but await the Phase II effort for testing. A stochastic model was developed to describe the distribution of glass fragments from an incident blast wave. Structural integrity for explosive environments will be obtained by using a commercially marketed hydrocode, AUTODYN™ and handbook solutions using the Task 3 environments as a forcing function.

In Task 5, a methodology was developed using the MRC *John O.* digital phantom. *John O.* will be interfaced with the Phase II virtual environment to describe the effect of penetrating wounds from the projectile environments described in Tasks 3 and 4. Tolerance to overpressure/blast effects will be described by threshold values of peak pressure and impulse mapped on to *John O.*

In Task 6, a digital "fly-through" model of the McKenna Town Hall was developed. Several approaches and codes were used to develop this model; including Multigen™, Division™, Coryphaeus™, and developing entirely new visualization tools. This was done to compare suitability of these approaches for visualizing weapon effects. Ultimately, we converged on a hybrid approach, incorporating "wrap-around" code to an existing commercial visualization tool. Static visual representations of bullet holes were also developed to understand the "choke" points in real-time visualization of these effects. Several approaches were also developed to model debris and ejecta.

The Phase I effort identified the key problems and relevant approaches for Phase II. Moreover, it attacked fundamental modeling and simulation issues which extend well beyond the current application; i.e., how to embed modeling of physical processes (which is necessary if the objects in a simulation are to interact in a non-scripted manner) in a real-time visual simulation.

PHASE I HIGHLIGHTS

An important theme embodied in this program was to not only represent physical processes engendered by weapon effects where *everything goes according to plan* and deterministic models are sufficient to describe outcomes but also to *incorporate analytical models in sufficient generality, with sufficient degrees-of-freedom, and sufficient detail where things that can and do go wrong can also be modeled and represented in the simulation.* For example, in Phase I, a small arms model was programmed which includes deterministic and stochastic modules.

The deterministic small arms model developed for the first phase of this project describes the *exterior* and *terminal ballistics* of rifle, hand gun, and shotgun ammunition striking various types of interior building constructions.

For rifle and hand gun ammunition, the non-linear trajectory through air, including cross wind effects (which can be significant in urban environments where wind is channeled), air drag, and gravity; as well as striking velocity, obliquity, and hit location on the target, are described. Projectile striking conditions are then used in conjunction with a library of projectiles (single slug, and two buckshot loads for a 12 gauge shot gun, 7.62 and 5.56 FMJ rifle and 9-mm parabellum handgun ammunition) and a library of interior building constructions (window, wall, ceiling, floor, and interior door cross sections) to determine subsequent trajectory after interaction with these targets (i.e., ricochet, penetration, or perforation). For shot gun loads with buckshot, in addition to the variables described above, shot dispersion as a function of range is calculated.

The stochastic portion of the model which can be toggled "on/off" with various options uses the deterministic *target hit point* calculation to describe the center of a random distribution of target striking locations. The distribution about this center is described by a *Circular Error Probability* (CEP). The CEP is obtained from experimental data bases which tabulate delivery errors in terms of shot dispersion and offset from an aimpoint. These data bases, which are specific to each weapon and projectile, have been collected by the Army to determine the effect of target exposure time, training, stress, and firing conditions (first shot, single shot, burst or automatic mode) on gunner accuracy (see Table 2 below for an example relative to target exposure time using a bipod mount).

The *external ballistics*, deterministic portion of the model, has been correlated with data supplied by the Army and ammunition manufactures (see Figure 1 and Figure 2, page 10, for correlation with 9-mm M9 Pistol Data and 5.56-mm M16A1 Rifle, respectively; data supplied in FM 101-60-27-1).

Table 2. Delivery Errors for 5.56-mm Rifle, M16A1[1]

Range (m)	First-Round Delivery Errors		Average Offsets[2]		Dispersion of Offsets		Subsequent-Projectile Dispersion	
	σ Horizontal (mils)	σ Vertical (mils)	σ Horizontal (mils)	σ Vertical (mils)	σ Horizontal (mils)	σ Vertical (mils)	σ Horizontal (mils)	σ Vertical (mils)
Standing Attacking Firer: 1-Second Target Exposure								
20	23.0	23.0	14.0	56.8	15.6	22.7	23.0	36.2
40	18.0	18.0	14.0	56.8	15.6	22.7	23.0	36.2
60	15.0	15.0	14.0	56.8	15.6	22.7	23.0	36.2
80	11.0	11.0	14.0	56.8	15.6	22.7	23.0	36.2
3- to 5-Second Target Exposure								
20	11.2	11.2	14.0	56.8	15.6	22.7	23.0	36.2
40	7.8	7.8	14.0	56.8	15.6	22.7	23.0	36.2
60	6.9	6.9	14.0	56.8	15.6	22.7	23.0	36.2
80	6.2	6.2	14.0	56.8	15.6	22.7	23.0	36.2

[1] *Bipod mount used.*
[2] *First-round biases assumed to be 0.0 if the gunner is zeroed in at the given range.*

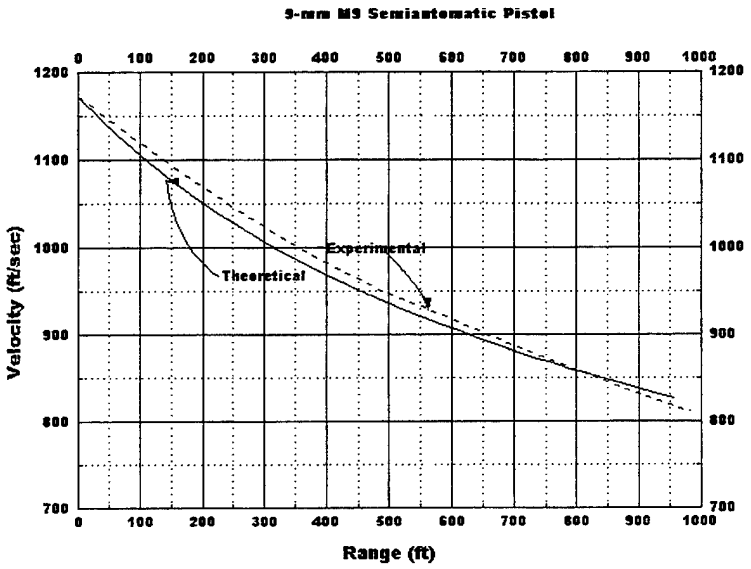


Figure 1. Projectile Striking Velocity as a Function of Target Range for the M9, 9-mm, Semiautomatic Pistol

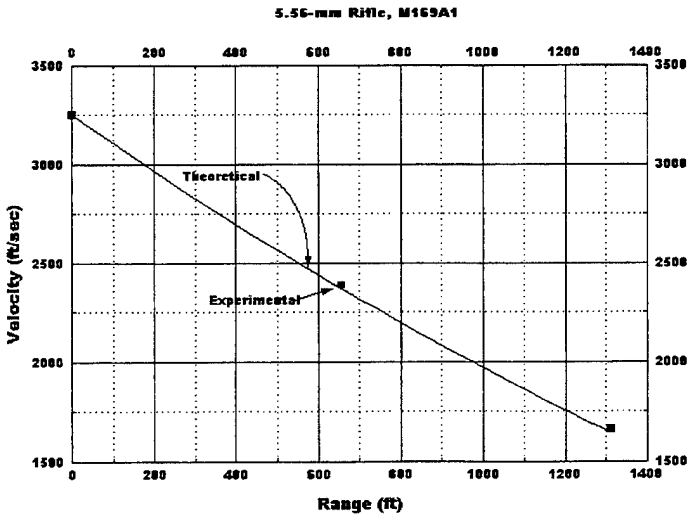


Figure 2. Projectile Striking Velocity as a Function of Target Range for M16A1 Rifle using 5.56-mm, M193, Ball Ammunition

The *terminal ballistics* portion of the deterministic model reflects ballistic data generated during Phase I of this effort. The projectiles, target cross sections, and results are shown in Table 3, Figure 3, and Table 4, respectively.

Although no ricochet data was available during the first phase of this effort, a "first principles" model originally developed to describe projectile ricochet from a human femur in soft tissue (see Figure 4 and Figure 5) as part of a current DARPA/MRC effort was modified to describe ricochet from building materials. A parametric form of this model has been incorporated in the Phase I software. The ricochet model will be correlated with experimental data developed during Phase II.

A comprehensive description of a MOUT task involving common subtasks of other MOUT tasks; i.e., clearing a building, was developed. In developing this task, MRC consulted with the U.S. Army Infantry School (USAIS) at Fort Benning, the FBI Academy in Quantico, Virginia, Training officers at the Los Angeles Police Department, and various Government Training manuals relative to Military Operations in Urban Terrain (MOUT). A building clearing operation conducted on the three-story Town Hall building at the Fort Benning, McKenna MOUT site was selected as the example scenario. The Town Hall building

Table 3. Ammunition used during STRICOM Ballistic Tests against Targets Representative of Residential Construction

Gun	Round	Total Mass (grams)	Diameter (inches)	Projectiles in Salvo	Muzzle Velocity (fps)
12 ga Shotgun	Rifled Slug	28.4	0.73	1	1600
12 ga Shotgun	#4 Cu-plated Buckshot	50.5	0.24	41	1210
12 ga Shotgun	#00 Lead Buckshot	29.6	0.33	9	1325
9mm Handgun	FMJ Parabellum	7.5	0.36	1	1155
7.62mm Rifle	FMJ	8.1	0.31	1	2100

has a good arrangement of rooms, interior walls, stairways, windows, and doors in which to demonstrate many of the problems, direct and collateral weapon effects, associated with using various weapons in a MOUT environment.

Table 4. Results from STRICOM Phase I Ballistic Experiments

SHOT #	TARGET TYPE	ROUND	MUZZLE VELOCITY	RESIDUAL VELOCITY	VELOCITY LOSS
01	INT. WALL	SLUG	1600	1500	6%
02	INT. WALL	#00-B	1325	1150	13%
03	INT. WALL	#4-B	1210	1090	10%
04	INT. WALL	9mm	1155	1100	5%
05	INT. WALL	7.62mm	2100	2080	0%
06	INT. DOOR	#4-B	1210	1050	13%
07	INT. DOOR	#00-B	1325	1080	18%
08	INT. DOOR	SLUG	1600	1470	8%
09	FLOOR	#00-B	1325	980	26%
10	FLOOR	SLUG	1600	1480	8%
11	FLOOR	#4-B	1210	950	21%
12	FLOOR	9mm	1155	1050	9%
13	CEILING	9mm	1155	1090	6%
14	CEILING	#4-B	1210	700	42%
15	CEILING	#00-B	1325	650	51%
16	CEILING	9mm	1155	1070	7%
17	CEILING	SLUG	1600	1470	8%
18	1/8" STD GLASS	#00-B	1325	N/A	N/A
19	1/4" PLATE GLASS	#00-B	1325	N/A	N/A

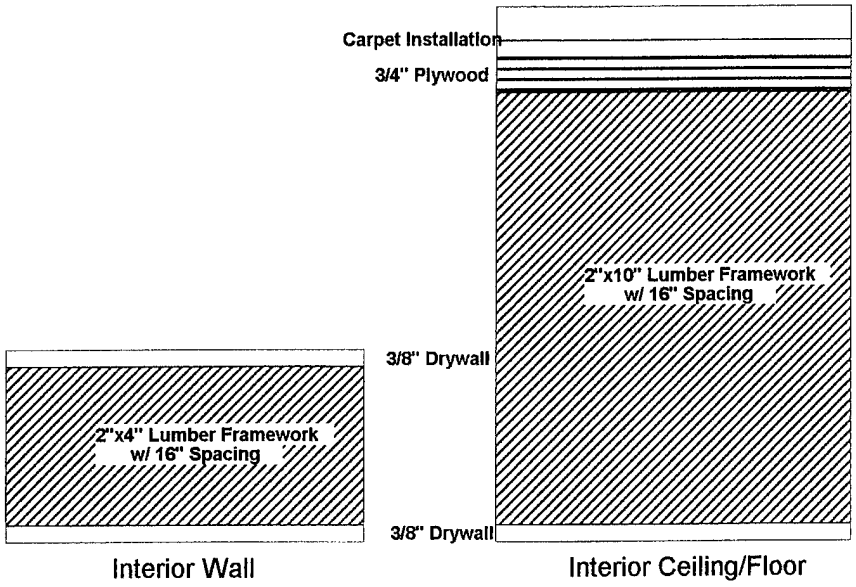


Figure 3. Phase I Ballistic Target Cross Sections

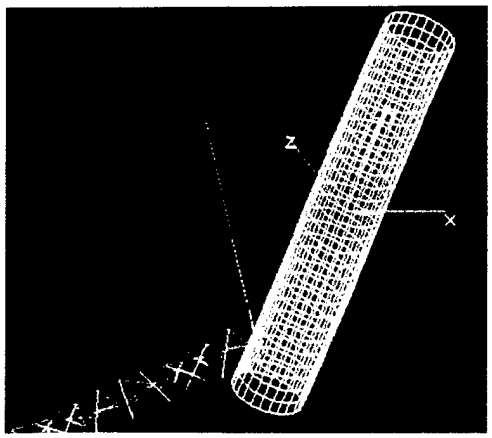


Figure 4. Frame from Animation of Glancing Strike by AK47 Projectile on Idealized Human Femur Imbedded in Gelatin Matrix (DAMD17-94-C-4099)

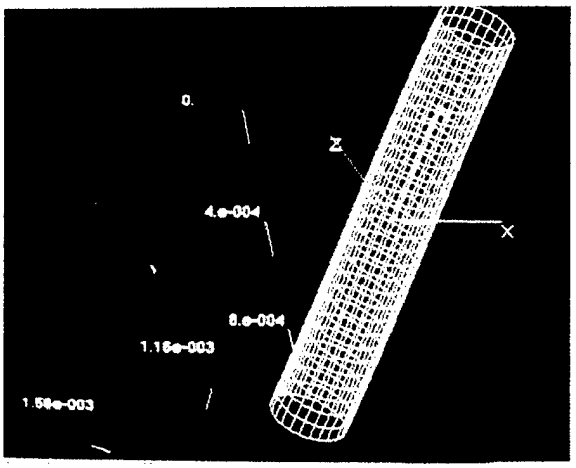


Figure 5. Elapsed Time in Microseconds of Glancing Strike by AK47 Projectile on Human Femur (DAMD17-94-C-4099)

Table 5. Types of Buildings and Frequency of Occurrence World-Wide

TYPE OF BUILDING	FREQUENCY OF OCCURRENCE (Percentage)
30-inch Stone	1
8- to 10-inch Reinforced concrete	6.9
12- to 24-inch Brick	63
6-inch wood	16
14-inch steel and concrete (heavy clad)	2
7-inch steel and concrete (light clad)	12

The initial set of Phase I domestic weapons considered was:

- M16A1, .22 caliber rifle firing 5.56-mm, M193 Ball rounds;
- M16A2, .22 caliber rifle firing 5.56-mm, M855 Ball rounds;
- M249, Squad Automatic Weapon (SAW), 22 caliber machine gun firing 5.56-mm, M855 Ball rounds;
- M60, .30 caliber machine gun firing 7.62-mm, M80 ball rounds
- 9-mm Semiautomatic M9 pistol firing M822 ball rounds
- 12-gauge "sawed-off" shotgun firing single slug, 00 and number 4 buckshot (not standard military issue); and,
- 40-mm M203 Grenade Launcher firing M406 HE and M433 HEDP rounds.

Three soldiers composed a fire team, and at least two fire teams were involved in a clearing operation.

It was observed that construction details of typical buildings, as well as corresponding weapon effects, can vary significantly from the cinder block and poured concrete construction of the McKenna Town Hall building (see Table 5). For this reason alternative, but more typical, wall, ceiling, floor, glazing, and door constructions were considered in Phase I. In Phase II, a library of user prescribed construction details will be included so that construction details of building components can be varied by the user during the simulation. The software will prompt the user for these details which should be the same considerations that occur to the user before employing weapons in a real MOUT mission. Weapon effects algorithms will describe differences in effects due to variation of construction methods and materials.

During Phase I Ballistic tests employing weapons and ammunition above on typical interior wall, ceiling, floor, door, and glazing materials were conducted at a range of eight feet. Exit velocity of the striking projectile through the rear surface of the target was determined, debris fields were video taped, ejecta was captured in witness plates, and the target entrance and exit holes were photographed. These tests were conducted

Table 6. Real Time versus Execution time for Software Algorithms in Small Arms Model

Muzzle Velocity (fps)	Range (ft)	Elapsed time to strike target after weapon fired (sec)	Execution time on 66 Mhz 80486 PC (sec)
1000	10	0.105	0.01
1000	100	0.105	0.02
2000	10	0.0052	0.01
2000	100	0.052	0.01

to capture phenomenology, yield preliminary data on projectile velocity loss after target penetration, and provide texture maps for the Task 6 Visualization and Scene Animation task.

Software, which executes in sub-real-time (see Table 6) was also developed to describe the *exterior* and *terminal ballistics* (including the effect of cross-wind, ricochet, and penetration) for handgun, rifle, and shotgun fired projectiles of interest (shot dispersion as a function of range is also described for shotgun rounds). Single shot, burst, and automatic modes were included in the model as well as the effect on aimpoint accuracy (in terms of circular error probability as a function of range) of:

- target exposure time;
- soldier stress;
- training; and,
- typical shot dispersion relative to an aimpoint.

A stochastic model was also developed to describe distribution of glass shards (in terms of ejecta mass and velocity) from small arms and blast waves. An approach which will be implemented in Phase II (described in the Phase II proposal) was also developed to describe the initial shock, hole size, and time and spatial evolution of overpressure from the blast and fragmentation from grenades and other larger ordnance (such as demolition satchels).

The effect of penetrating wounds on personnel is accomplished using the "John O." phantom developed by MRC in a U.S. Army Natick Phase II SBIR (DAAK60-C-92-008) – see Figure 6, Figure 7, Figure 8, and Figure 9. This digital representation of a soldier divides the human anatomy into 80,000 volume elements and 250 tissue types. The anatomy can be articulated and scaled to vary anthropometry. Currently, body armor and ballistic protective goggles can be placed on *John O.*, dynamic exposure probabilities of user prescribed regions on the human body during task performance

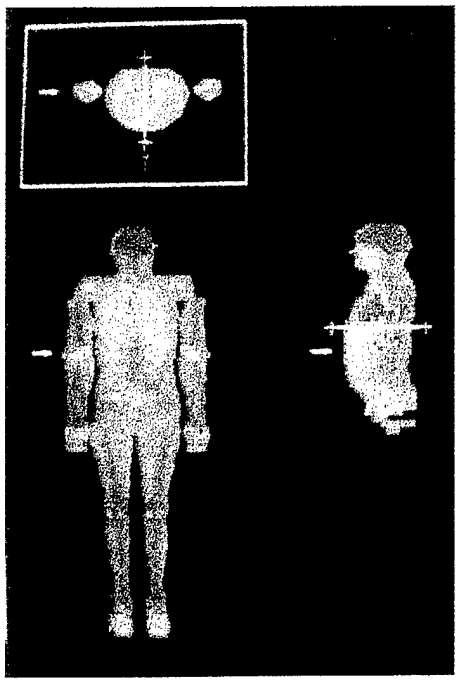


Figure 6. Autopsy Mode of "John O." (MRCMAN) Phantom

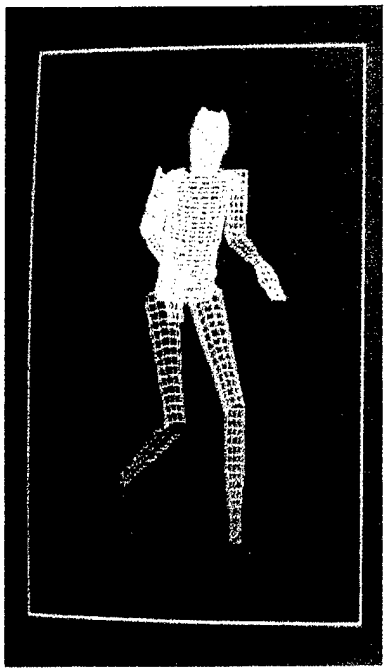


Figure 7. Projection Mode of John O

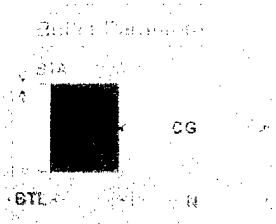


Figure 8. Screen from Bullet Menu

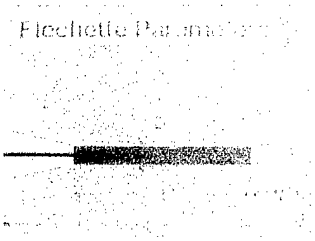


Figure 9. Screen from Flechette Menu

can be obtained, and the effect of penetrating wounds from virtual small arms can be demonstrated on *John O*. As of September 1996, *John O*. has resulted in almost $2M of Phase III contract awards.

A digital representation of the McKenna Site Town Hall, that can be explored in real-time using a Silicon Graphics workstation was also developed. This was done to identify constraints associated with incorporating real-time representations of weapon effects in the Phase II virtual environment. A narrated video tape of a user exploring the MRC Digital Town Hall in real-time was sent under separate cover to STRICOM (Admiral Piper).

2. SMALL ARMS EXTERIOR AND TERMINAL BALLISITCS MODEL

This section discusses the analytical and numerical background of the MRC developed computer code, *MRSAW* (Mission Research Small Arm Weapons). A listing of the MRSAW source code is presented in Section 5 of this report. This code was developed during the first phase of this effort and is capable of performing the analyses listed below in sub-real-time. A *Nomenclature* describing all variables and parameters used in Section 2 is presented in Table 7, page 17.

I. EXTERIOR BALLISITCS. MOTION OF BULLETS IN AIR PRIOR TO HITTING THE TARGET UNDER PRESCRIBED INITIAL CONDITIONS, I.E., MUZZLE VELOCITY, ANGLE MADE BY THE INITIAL BULLET PATH WITH THE HORIZONTAL, AND/ OR TARGET LINE-OF-SIGHT

II. TERMINAL BALLISITCS

IN THE CASE OF TARGET PENETRATION, MRSAW CALCULATES

- PROJECTILE STRIKING VELOCITY ON THE TARGET
- PROJECTILE EXIT VELOCITY FROM THE TARGET
- TOTAL ENERGY DEPOSITED ON THE TARGET (RELATED TO CREATION OF DAMAMGE)
- DAMAGE AREA OF THE TARGET IN THE CASE OF BUCKSHOT OR SLUGGER ROUNDS FIRED FROM A SHOTGUN

IN THE CASE OF NO PENETRATION, MRSAW CALCULATES

- BULLET RICOCHET
- PROJECTILE MOTION AFTER REBOUND

Target locations and material properties need to be specified by the user in order to implement calculations in items (II) and (III),

The first phase of projectile motion, referred to as projectile *exterior ballistics*, describes the projectile trajectory, prior to impact, subject to gravity, aerodynamic drag, and cross wind effects. Projectiles fired from rifles and other small arms as well as projectiles launched from fragmenting munitions (grenades, artillery shells, and bombs) are considered. If aerodynamic drag can be specified exactly, the problem of determining the projectile trajectory is deterministic even though the resulting equations of motions may be nonlinear. Under these circumstances, an exact closed form solution for the projectile motion is not possible.

Table 7. Section 2 Nomenclature

SYMBOL	MEANING	WHERE DEFINED
A	Velocity modified drag coefficient	Page 23, equation (6a)
a	acceleration	Page 20, equation (1)
$C(S_c)$	Ballistic coefficient	Page 20
D	Lateral projectile dimension	Page 31
D_i	Projectile diameter in inches	Page 21, equation (3)
d	Target thickness	Page 29
E_{cr}	Minimum energy needed for complete target penetration for a given projectile-target pair	Page 30
e	Coefficient of restitution between a given projectile-target pair	Page 33
F_0	Defined constant $(=\frac{v_0^n}{A})$	Page 24
F_r	Constant force of retardation during target penetration	Page 29
g	Acceleration due to gravity	Page 23
L	Longitudinal projectile dimension	Page 31
N	Total number of pellets in a shot	Page 34
m	Projectile mass	Page 29, equation (12)
n	$=n(v)$ from Ingall's Table 8	Page 21
$n(v)$	Power index: Mayevski's parameter	Page 20, equation (2)
$n(R,r)$	Number of pellets within radius r at range R	Page 33
$n1,n2$	Experimental data points for $n(R,r)$	Page 34
R	Range in feet	Table 10, Page 25
$R1,R2$	Experimental points for range R	Page 34
R_f	Reference range for experimental data	Page 34, equation (15)
r	Radial coordinate on the target plane	
r_{max}	Spray radius for damage on the target	Page 39
S_c	Shape class parameter	Page 21
v	Projectile speed	Page 19
v_0	Muzzle velocity	Page 19
v_{in}	Incident projectile velocity on target	Page 29, equation (13)
v_{ex}	Projectile exit velocity after target penetration	Page 29, equation (13)
v_{cr}	Minimum incident velocity needed for complete target penetration for a given projectile-target pair	Page 30
W_i	Projectile mass in grains	Page 21, equation (3)

Nomenclature-Continued

SYMBOL	MEANING	WHERE DEFINED
$x(t)$	Horizontal location of the projectile at time t	Page 22
$\dot{x}$	Projectile speed along the x-direction	Page 23
$y(t)$	Vertical location of the projectile at time t	Page 22
$\dot{y}$	Projectile speed along the y-direction	Page 24
$z(t)$	Lateral location of the projectile at time t	Page 22
$\dot{z}$	Projectile speed along the y-direction	Page 24
$\dot{\alpha}$	overhead dot indicates time derivative of any α	Section 2
$\alpha(v)$	Meyevski's parameter	Page 20, equation (2)
$\beta(v)$	Air drag factor due to compression	Page 20, equation (1)
δ	Shooting angle with the horizontal	Page 22, Figure 10
λ	Model parameter for $n(R,r)$	Page 34, equation (15)
μ	Model parameter for $n(R,r)$	Page 34, equation (15)
θ	Spray angle for pellets	Page 34

2.1 Rationale Behind Using the Full, Nonlinear, Coupled Theory to Describe Projectile Motion

When a projectile is launched in space either as a bullet from a gun or as a fragment from a blast, its trajectory in space depends on the following geometric and kinematic properties:

1. Initial speed at the time of its release in air;
2. Initial direction of its velocity vector at the time of its release in air;
3. Air drag coefficient which is the constant of proportionality in the direct variation of projectile deceleration with the square of the instantaneous projectile speed (This is in accordance with the Newton's law of aerodynamic drag); and,
4. Gravitational acceleration due to the weight of the projectile.

Size and shape control rotational motion about the projectile center of mass while the parameters listed above (items 1 through 4) control the trajectory of the projectile center of mass.

For cases where the initial velocity of the projectile is very high (more than 100 ft/sec), and short flight paths are considered (50 to 100 ft), gravitational effects can be ignored so that the projectile, idealized as moving in a straight line, yields acceptable results.

Strictly speaking, under any circumstances, when gravity is present and the projectile is not moving along a vertical line, the trajectory of the center of mass of a projectile is curved so that its curvature, $\frac{1}{\rho}$, mass, m, and speed, v, is related to the normal component, F_n, of forces resulting from the air drag and gravity according to Newton's second law of motion, viz., $F_n = m\frac{v^2}{\rho}$. According to this equation, the issue of whether the assumption of a straight line path of the projectile is valid or not, really hinges on F_n. This result can be put in a better perspective by writing this equation as $\frac{a}{\rho} = \frac{F_n a}{mv^2}$ where a is a characteristic dimension of the projectile. Thus, curvature can be ignored when $\frac{F_n a}{mv^2} << 1$. With decreasing projectile velocity, this condition ceases to be valid after some point in time (see Figure 10).

In order to develop a code which is capable of predicting projectile location and velocity at any time after it is launched in space, we can not in general depend on a model which relies on a linear projectile path. That is, we want the ability to describe projectiles launched from building interiors to the outside as well as the ability to describe projectiles launched from the outside into building interiors. We also want the ability to model collateral effects which may involve debris and fragments ejected over large distances. These situations may involve long flight paths as well as large forces retarding projectile motion which violate the condition above. In the absence of channeled air flows, linear assumptions are probably acceptable for projectiles launched within building interiors at interior targets since these involve high projectile velocities and short flight paths.

If we assume that the aerodynamic drag on the projectile is known as function of the

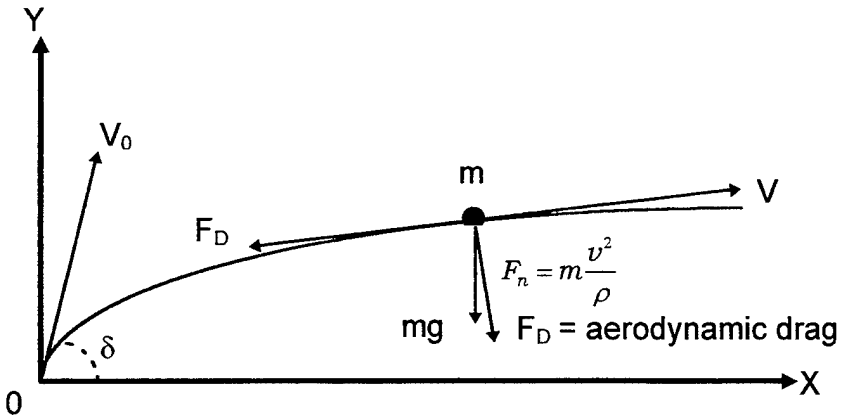


Figure 10. Effect of Normal Forces on Trajectory Curvature

instantaneous projectile speed, then the resulting equations of motion, in the presence of gravity, yield three (for full 3-D problem) or two (for a two-dimensional problem) coupled, nonlinear second order differential equations. These equations can be numerically solved by using a modified version of the Runge-Kutta intergration algorithm for solving nonlinear, second order, differential equations.

For high projectile velocities and short flight paths, these equations reduce to uncoupled, nonlinear, second order differential equations. These equations possess complete analytical solutions. The numerical code developed under this program yields numerically integrated results from the coupled theory and analytical solutions from the uncoupled theory. For most applications, execution times to obtain both solutions are comparable. However, if results from the uncoupled equations are acceptable, then this solution should be used when the code is used repeatedly to analyze a series of these types of problems (e.g., grenade fragments and automatic rifle fire inside a room).

According to conventional formulations of aerodynamic drag, the force of retardation or drag acting on a projectile moving in air or fluid with a speed, v, is directly proportional to the projected cross-sectional area of the projectile on a plane normal to the direction of the velocity of the projectile and the square of the speed, v, i.e., v^2. However, depending on the range of v, the constant of proportionality may depend on the speed, v. Using the above law of aerodynamics, the deceleration $-a$ of the projectile can be written as:

$$-a = \beta(v)v^2; \tag{1}$$

where $\beta(v)$ is related to the Meyevski's parameter $\alpha(v)$ and index $n(v)$ through the equation:

$$\beta(v) = \frac{\alpha(v)}{C(S_C)} v^{-n(v)} \tag{2}$$

In Equation (2), $C(S_c)$ is the ballistic coefficient[4] which is a function of the bullet shape class, S_c. The concept of a bullet shape class that influences retardation of a projectile in air stems from development of a reference projectile with which other projectiles are scaled. For our purposes, the reference projectile has a weight of one pound, a diameter of one inch, and a ballistic coefficient, $C = 1$. For the reference projectile, the dependency of the Meyevski's parameter, $\alpha(v)$, and index, $n(v)$, on the instantaneous speed, v, of the projectile is shown in Ingall's table (Table 8).

Ingall's table indicates that for low projectile velocities ($v < 790 fps$) n=0 so that the retardation formula presented in Equation (1) reduces to the classical Newton

[4] A. J. Pejsa, MODERN PRACTICAL BALLISTICS, Second Edition, Kenwood Publishing, 1989.

Table 8. Ingall's Table: Retardation coefficients α(v) and n(v) for various velocities, v, for the reference projectile.

α(v)	n(v)	v (fps)
4.065 E-3	0.45	2600 < v < 3600
1.248 E-3	0.30	1800 < v < 2600
0.01316 E-3	0	1370 < v < 1800
0.957 E-7	-1	1230 < v < 1370
0.6337 E-13	-3	970 < v < 1230
0.5935 E-7	-1	790 < v < 970
0.4676 E-4	0	0 < v < 790

Table 9. Shape Class Descriptions

Shape Class (S_c)	Projectile Shape
1	Slender Spitzer Boat tail
2	Flat Base Spitzer
3	Semi-round nose or semi-Spitzer
4	Round nose or flat nose common in 30-30 cartridge

Note: In-between shape class may also be used in MRCSAW code.

formulation of aerodynamic drag with a v^2 dependency. For higher velocities, the power index of v as well as Meyevski's parameter depends on projectile velocity due to increasing air compression (air density) in front of the projectile with increasing projectile velocity.

The algorithm used in MRSAW code to calculate the parameters *α(v), n(v), and C(S_c)* is as follows. Ingall's table is included in the database as a lookup table. For any input muzzle velocity, these parameters are read from the table, and are used in the drag calculation. For the determination of the ballistic coefficient, $C(S_c)$, for a given projectile, we use a shape class parameter, S_c, which is an index parameter that describes the shape of the projectile. The index for the shape parameter S_c is shown in Table 9.

Using a well established empirical relation, the indexed shape parameter S_c is used to calculate the ballistic coefficient, *C*, for any given projectile. This relation is given by

$$C(S_C) = \frac{W_t\left(\frac{D_i + 0.5}{D_i^2}\right)}{415(8 - 2S_C + S_C^2)} \qquad (3)$$

In equation (3), W_t is the projectile mass in grains (1grain=0.0648 gram) and D_i is the projectile diameter in inches.

2.2 Determination of the First Phase of Motion in Exterior Ballistics: Equation of Motion of a projectile moving through air

The kinematics of a projectile moving in air is shown in Figure 11. Assuming that the bullet of mass *m* is moving in the xy-plane, Newton's second law of motion yields the following equations of motion along the x- and y-axis respectively.

$$\ddot{x} = \frac{-\alpha(v)v^{2-n(v)}}{C(S_C)}\left(\frac{\dot{x}}{v}\right) \tag{4a}$$

$$\ddot{y} = -g - \frac{\alpha(v)v^{2-n(v)}}{C(S_C)}\left(\frac{\dot{y}}{v}\right) \tag{4b}$$

$$v = \sqrt{\dot{x}^2 + \dot{y}^2}\ ;\ z(t) \equiv 0$$

In Equation (4), we have used overhead dots to indicate time derivatives.

From the nature of the differential equations in (4), it is obvious that they are nonlinear and hence can not be integrated in closed form to express the velocity and location of the projectile as a function of time.

2.3 Linearization of Equation (4)

Under some restrictive cases, it is possible to linearize Equation (4) so that a closed

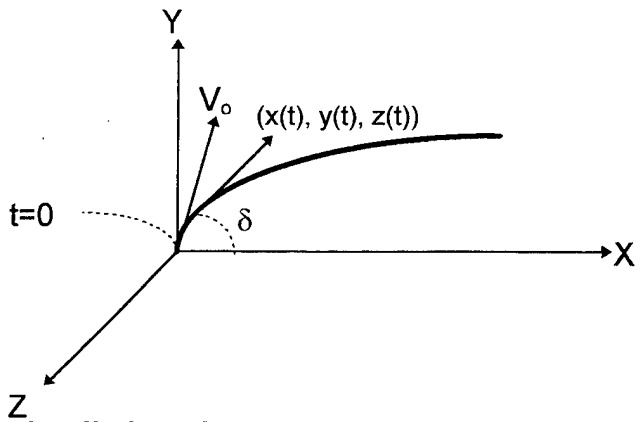


Figure 11. Projectile location at time t

form solution for the projectile in the hodograph (locus of speed as a function of location) in the physical plane can be derived. The rationale behind such linearization is discussed below.

For bullets shot at high speeds (> 1000 fps) at low angles (< 5 degrees) for short ranges (< 100 ft.), the bullet path is mostly horizontal so that the velocity along the horizontal direction is much larger than the velocity in the vertical direction. Besides, no significant reduction in velocity occurs during the travel time. In view of these considerations, the following assumptions can be justified.

With reference to Figure 11, we assume that the bullet location at any time t is given by the coordinates $\{x(t), y(t), z(t)\}$ where, in the absence of any cross-wind in the z-direction, $z(t) \equiv 0$. At the end of this section, we will discuss the effect of cross-wind along the z-direction. The exact equations of motion (4) can be rewritten as [Reference. 4].

$$\ddot{x} = \frac{dv}{dt} = v\frac{dv}{dx} = \frac{-\alpha(v)}{C(S_C)} v^{2-n(v)} \left(1 + \frac{\dot{y}^2}{\dot{x}^2}\right)^{-\frac{1}{2}} \approx \frac{-\alpha(v)}{C(S_C)} v^{2-n(v)} \qquad (5a)$$

$$v = (\dot{x}^2 + \dot{y}^2)^{\frac{1}{2}} \approx \dot{x}^2 \text{; for } \frac{\dot{y}^2}{\dot{x}^2} << 1$$

$$y = -g - \left(\frac{\alpha(v)}{C(S_C)}\right) \dot{y} v^{1-n} \qquad (5b)$$

Assuming that in the range of v considered, $\alpha(v)$ and $n(v)$ are constants so that for the reference projectile ($C=1$), Equation (5) becomes

$$\ddot{x} = \frac{dv}{dt} = v\frac{dv}{dx} = -Av^{2-n} \text{; where } A = \alpha(v)/C(S_C) \qquad (6a)$$

$$\ddot{y} = -g - A\dot{y}v^{1-n} \qquad (6b)$$

In Equations (6), $\alpha(v)$ = *constant* = A and $n(v)$ = *constant* = n whose values can be computed from Ingall's Table, Table 8, for a reference projectile.

Integrating (6a), yields:

$$\dot{x} = v = v_0 \left(1 - \frac{nx}{F_0}\right)^{\frac{1}{n}} \qquad (7)$$

In Equation (7), we have used the initial condition that $\dot{x}(0) = V_0$ *and* $F_0 = \frac{V_0^n}{A}$.

Defining a function $F(t) = Av^{-n}$, we see that $F(0) = F_0$ *and* $F = F_0 - nx$

Integrating (7) again with respect to *t*, yields:

$$x = \frac{F_0}{n}\left[1 - \left\{1 + \frac{(1-n)v_0 t}{F_0}\right\}^{\frac{n}{n-1}}\right] \tag{8}$$

Similarly, integrating (6b), we have

$$\dot{y} = \frac{\dot{x} g F_0}{(n-2)v_0^{\,2}}\left\{\left(1 - \frac{nx}{F_0}\right)^{1-\frac{2}{n}} - 1\right\} \tag{9}$$

where we have used the initial condition $\dot{y}(0) = 0$ (small angle approximation).

Integration of Equation (9) again, yields:

$$y = \frac{gF_0}{(n-2)v_0^{\,2}}\left\{\frac{F_0}{2(1-n)}\left[\left(1 - \frac{nx}{F_0}\right)^{2-\frac{2}{n}} - 1\right] - x\right\} \tag{10}$$

In MRSAW, we have used equations (7)-(10), to calculate the velocity and location of the projectile at any time t (or any range *x*=*R*) when the initial muzzle conditions are given. This solution has been referred to as 'Linear Theory Results' in MRSAW.

2.4 Some Drawbacks of the Linearized Solution

From the initial conditions used in deriving the linearized solutions (7)-(10), it is obvious that the solution is independent of the initial angle of shooting δ with the horizontal (Figure 11). As this angle significantly influences the rise and drop of the bullet from the horizontal line, solutions presented by equations (7)-(10) can not be expected to be accurate in estimating the rise and drop of the bullet for any given δ. This can be verified later when the solution from the exact equations are presented. Besides, the basis of the linearization process is dependent on the accuracy of the condition that

Table 10. Comparison between linearized solution and exacy solution for the reference projectile with an initial velocity of 3,000 fps

Angle, δ (Degrees)	Range, R (feet)	Exact Soln. For Impact Velocity (fps)	Linear Soln. For Impact Velocity (fps)	Exact Soln. Rise (+) or Drop (-) in inches	Linear Soln. Rise (+) or Drop (-) in inches
0	10	2997	2997	- 0.002	- 0.002
0	25	2992	2992	- 0.0124	- 0.0124
0	50	2984	2984	- 0.054	- 0.054
0	100	2968	2968	- 2.15	- 2.15
0	1000	2687	2687	- 22.57	- 22.48
1	10	2997	2997	2.02	2.01
1	25	2992	2992	5.02	5.02
1	50	2984	2984	10.41	10.41
1	100	2967	2968	20.65	20.66
1	1000	2686	2687	184.32	184.08
2	10	2997	2997	4.03	4.03
2	25	2992	2992	10.05	10.05
2	50	2984	2984	20.86	20.88
2	100	2967	2968	41.52	41.52
2	1000	2686	2687	391.20	390.94
5	10	2997	2997	10.49	10.09
5	25	2992	2992	26.15	25.20
5	50	2984	2984	52.20	52.40
5	100	2967	2968	104.0	104.41
5	1000	2686	2687	1010.76	1012.92

$\left(\frac{\dot{y}}{\dot{x}}\right)^2 << 1$ which becomes a bad approximation as the shooting angle δ with horizontal increases or when the range increases.

For applications in buildings, range is not really an issue since the target range is below 100 ft. for most applications, but the shooting angle can be arbitrary since either the wall, or the floor or the ceiling may be targets.

2.5 Exact Solution of the Projectile Equation of Motion

The exact solution of the problem of describing the trajectory of the bullet can be obtained from the differential Equation (4) under the general initial conditions (Figure 11):

$$\begin{aligned} \dot{x}(0) &= v_0 \cos\delta \\ \dot{y}(0) &= v_0 \sin\delta \\ x(0) &= 0 = y(0) \end{aligned} \qquad (11)$$

Since the differential equations in (4) are nonlinear, no closed form solution can be obtained as in the case of the linearized form of these equations shown in (5). Solution has been obtained from an extension of the highly accurate algorithm for solving one-dimensional, nonlinear, second order differential equations known as the Runge-Kutta method (RKM). The extended version of this algorithm can solve either two (for planar projectile motion) or three (for fully three-dimensional projectile motion) coupled nonlinear differential equations. The validity of this extended version of RKM has been verified by comparing the solution from RKM with exact solutions for coupled, second order differential equations where exact solutions are available. This version of the solution is referred to as 'exact solution' in MRSAW.

Some effect of the initial angle δ on the linearized solutions (7)-(10) can be incorporated as follows. The rise of the bullet in the absence of any vertical deceleration due to gravity is given by $y = x\tan(\delta)$ so Equation (10) can be modified by the addition of this term to the right hand side of (10). This modification has been included in MRSAW.

Some comparison of the results predicted by the exact and linearized solution of the bullet flight path can be made at this point. Let us consider a target at distance varying between 10 and 1,000 feet. For the reference bullet, results for a muzzle velocity of 3,000 fps and various shooting angles are presented in Table 10. It is clear from Table 10 that results are in good agreement for horizontal location of the bullet while vertical location predicted by the linear solutions are in error for shooting angles above 4 degrees.

For any application results from both types of solutions are presented. The logic to automatically select the appropriate solution will be included in the Phase II software. The rest of the MRSAW development described below employs the exact solution form of the equations of motion shown (Equations 4).

2.6 Second Phase of the Exterior Ballistics

The first phase of the exterior ballistics deals with the flight path of the projectile from the weapon to the target. At the time of hitting the target, MRSAW presents the data just before target impact. These data include: (1) Projectile striking velocity; and, (2) Projectile impact angle on the target.

What happens after the projectile hits the target depends on the material and geometrical properties of the projectile-target pair. Following are the possible scenarios.

In the case of a bullet, we may have:

1. Bullet incapable of penetrating the target and the force of impact may cause the bullet to ricochet after impact;
2. If ricochet occurs, the plane of motion of the bullet may differ from the plane of motion (for two-dimensional motion) prior to impact. The plane of motion after impact is the plane that contains the bullet line of impact and the line of force of impact from the target. In MRSAW, the line of force of impact between the projectile and the target is assumed to be normal to the target at the point of impact. For various projectile-target pairs of interest, the nature of the line of impact needs to be determined from experiments; and,
3. If the bullet penetrates, the exit velocity of the bullet is of interest (Terminal Ballistic phase).

In the case of buckshot evaluated during Phase I and launched from shot guns, ricochet does not occur (since the buckshot is made predominately from lead and will therefore tend to deform). These projectiles may however:

1. Rebound after hitting the target. In this case, its flight path after impact needs to be determined; and,
2. Penetrate the target in which case its exit velocity is of interest.

Since the solution for the post impact conditions depends on the projectile-target pair and their properties, we have used test data for various projectiles-targets combinations. MRSAW uses these test results to predict the exit velocity when the projectile penetrates the target. In the case of projectile rebound, the force of impact is calculated from an assumed coefficient of restitution which also needs to be determined from experiments during Phase II for various projectile target-pairs. For the determination of the second phase of exterior ballistics, MRSAW has the following projectiles and target properties in its data base. This data base is derived from the MRC experiments during the first phase of this program, and is presented in a tabular from in Table 11.

The library of projectiles currently included in MRSAW are listed below.
1. Shotgun Single Slug
2. Shotgun #00 lead spheres
3. Shotgun #04 copper plated lead spheres
4. 9-mm parabellum handgun bullet
5. 7.62-mm and 5.56-mm rifle bullets

The library of targets currently included in MRSAW are listed below.
1. Generic interior wall
2. Generic interior floor
3. Generic interior door
4. Generic ceiling

Table 11. Results from Phase I Ballistic Experiments

Shot Number	Target Type	Round	Shape	Mass (grams)	Dia. (inches)	Number of Projectiles	Muzzle Velocity (fps)	Residual Velocity (fps)	Energy Loss ($gm\text{-}ft^2/sec^2$)E6	Energy Loss Area Corrected ($gm\text{-}ft^2/sec^2$)E6
1	INT. WALL	SLUG	SPHERE	28.4	0.73	1	1600	1500	4.402	8.260461625
8	INT. DOOR	SLUG	SPHERE	28.4	0.73	1	1600	1470	5.66722	10.63467818
10	FLOOR	SLUG	SPHERE	28.4	0.73	1	1600	1480	5.24832	9.848601989
17	CEILING	SLUG	SPHERE	28.4	0.73	1	1600	1470	5.66722	10.63467818
2	INT. WALL	#00 LEAD	SPHERE	29.6	0.33	9	1325	1150	6.41025	58.86363636
7	INT. DOOR	#00 LEAD	SPHERE	29.6	0.33	9	1325	1080	8.72053	80.07832874
9	FLOOR	#00 LEAD	SPHERE	29.6	0.33	9	1325	980	11.76933	108.0746556
15	CEILING	#00 LEAD	SPHERE	29.6	0.33	9	1325	650	19.73025	181.177686
3	INT. WALL	#04 CU-PLA	SPHERE	50.6	0.24	41	1210	1090	6.9828	121.2291667
6	INT. DOOR	#04 CU-PLA	SPHERE	50.6	0.24	41	1210	1050	9.14848	158.8277778
11	FLOOR	#04 CU-PLA	SPHERE	50.6	0.24	41	1210	950	14.20848	246.675
14	CEILING	#04 CU-PLA	SPHERE	50.6	0.24	41	1210	700	24.64473	427.8598958
4	INT. WALL	9MM	CYLINDER Blunt Hemisphere	7.5	0.36	1	1155	1100	0.46509375	3.588686343
12	FLOOR	9MM	CYLINDER Blunt Hemisphere	7.5	0.36	1	1155	1050	0.86821875	6.69921875
13	CEILING	9MM	CYLINDER Blunt Hemisphere	7.5	0.36	1	1155	1090	0.54721875	4.222366898
16	CEILING	9MM	CYLINDER Blunt Hemisphere	7.5	0.36	1	1155	1070	0.70921875	5.472366898
5	INT. WALL	7.62MM	CYLINDER Sharp Ogival	8.1	0.31	1	2100	2080	0.33858	3.523204995

The properties of these projectiles and targets are given in Table 3, and Figure 3. In the case of penetration, exit velocity has been calculated using the results in Table 11 as follows.

Assuming that the retarding force, F_r, during penetration is a constant, the amount of energy loss due to the penetration is equal to the product of $F_r.d$ where *d* is the depth of the target in the direction of penetration. This result can be shown by integrating the one-dimensional equation-of-motion (assuming that the penetration process can be modeled as an equivalent one dimensional problem) of a projectile of mass, *m*,

$$m\dot{x}\frac{d\dot{x}}{dx} = m\ddot{x} = -F_r \tag{12}$$

Integrating Equation (12) from *x = 0* to *x = d*, we have

$$\frac{1}{2}m(v_{in}{}^2 - v_{ex}{}^2) = F_r d \tag{13}$$

In Equation (13), the subscripts '*in*' and '*ex*' refer to quantities just before and after penetration of the target. Since the left hand of Equation (13) represents energy loss, ΔE, due to penetration, Equation (13) indicates that column 10 of Table 11 is equal to the product $F_r.d$ of the average force of retardation, F_r, and the depth of penetration, d. Assuming that $F_r.d$ is a property of the projectile-target pair, Equation (13) can be rewritten as Equation (14) to yield the exit velocity, v_{ex}, when the entry velocity, v_{in}, is given.

$$v_{ex} = v_{in}\sqrt{1 - \frac{F_r d}{\frac{1}{2}m v_{in}{}^2}} \tag{14}$$

2.7 Determination of Bullet Ricochet and Slug Rebound

In the context of this report, *ricochet* refers to motion of the projectile which is described in terms of both motion of the projectile center-of-mass and rotation about the projectile center-of-mass. In other words, both the plane of the projectile motion and rotational kinematics changes as a result if the target interaction. Projectile *rebound* merely requires a description in terms of the projectile center-of-mass, i.e., the plane of the projectile motion changes as a result of the target interaction. Bullet ricochet and slug rebound are discussed in Sections 2.7.1 and 2.7.2, respectively.

2.7.1 Bullet Ricochet

Bullet ricochet occurs when the bullet does not have enough incident energy to penetrate the target. This threshold energy, E_{cr}, is dependent on the projectile and target materials as well as their shape properties, and can be established from experimental results. For a given projectile-target pair, this threshold energy can be converted to a threshold impact velocity, v_{cr}, below which no penetration is possible, and the bullet ricochets after impact. In MRSAW v_{cr} has been calculated from $v_{cr} = v_{in}$ such that the exit velocity is $v_{ex} = 0$ in Equation (14). This gives: $v_{cr} = \sqrt{\frac{2F_r d}{m}}$.

The ensuing motion associated with bullet ricochet is much more complicated than the initial projectile motion prior to target impact when no bullet ricochet occurs. Here we face another aspect of the bullet motion which is the rotation of the bullet about its center of mass, *G*. For a general slender body (such as a bullet) the kinematics and kinetics of the bullet during its ricochet is presented in another MRC reference.[5]

2.7.2 Slug Rebound Mode

This problem is analytically identical to the first phase of the projectile exterior ballistics, except that the initial conditions are adjusted according to the post impact condition of the bullet. The relevant post impact conditions are: (1) Rebound speed after impact; and, (2) Direction of rebound after impact. When these two initial conditions are calculated from the impact equations, equations similar to Equations (4) can be used to calculate the trajectory of the projectile after impact.

2.7.3 Coordinate Systems used in the Input/Output of MRSAW

All coordinates used in MRSAW are referred to the initial coordinate system used in Phase I of the exterior ballistics portion of the code and is shown in Figure 11. All output is in the same coordinate system for first and subsequent phases of projectile exterior ballistics.

2.7.4 Graphic Representation of the Results from MRSAW: Input and Output Files

A detailed description of the input and output files used in MRSAW is shown in Table 13. Descriptions of the parameters used in the input file given in the input file as

---

[5] Eisler, R.D., and Chatterjee, A.K., *Algorithm Development to Describe Retardation in Human Tissue of a 19.6 Grain Flechette*, Mission Research Corporation Report MRC-COM-R-93-0372, US Army Natick Technical Report, March 1993 (Unclassified - Limited Distribution).

comments to the right of the numerical input along with the units used in the code. A typical input file is shown below in Table 12.

2.7.5 Buckshot: Area of damage

In the case of buckshot, a prescribed number of spherical projectiles are launched with the same initial muzzle velocities but with some velocity components in the z-direction (Figure 11). As they travel, these spherical projectiles cover areas proportional to the horizontal distance traveled. The constants of proportionality for particular buckshot is included in the MRSAW database, and can be used to determine damaged area of the target for this type of ammunition. For the second phase of the exterior ballistics, these results are shown on the screen when relevant during the execution of MRSAW. The impact locations on the target are specified in MRSAW in cylindrical coordinates. The specific radial and angular coordinates, in the target plane, of each striking projectile is determined by a random number which is generated consistent with the pellet distribution function, $n(R, r)$, defined in Equation (15), Section 2.7.9.

Table 12. Typical Input file for MRSAW code

```
3000.0,1.0,1200.0          INITIAL SPEED(FT/S),ANGLE(DEG),RANGE(FT)
30.0,10.0                  CROSSWIND SPEED(FT/S) IN THE HORIZ. & VERT. DIR.
10.0,5.0                   NORMAL TGT. ROT.(DEG) WRT. Z-AND-Y- OF TGT. AXES
0.30                       COEF. OF RESTITUTION FOR BULLET-TARGET IMPACT
1.0,1.0,1.0,1.0,1.0        BULL. DEN., MASS(GM), RADIUS, BODY&TAIL LENGTH(FT)
0.5,0.5                    FRONT AND REAR TIP DISTANCE FROM CM G (FT)
1.0,1.0                    BULLET MI ABOUT G (GM*FT**2) AND DIST. G FROM TIP
```

2.7.6 Fragmenting Munitions

Fragments generated during bomb explosions have less deterministic structures than the types of projectiles discussed before. Some of the problems stem from the fact that the size and shapes of these fragments can at best be described by probabilistic distribution functions. If a particular fragment is identified by 'the so called *L/D* ratio', which is the ratio of the longitudinal dimension, *L*, to the lateral dimension, *D*, then the distribution function of this parameter is available in the literature. This distribution function describes the probability of *L/D* to lie within a given domain. In order to use this information to access the damage produced by these bomb fragments, we need to establish either the deterministic or probabilistic distribution of the projectile trajectories. It is obvious from the deterministic analysis presented in the preceding sections for the trajectories of bullets and slugs, that we need to know the initial conditions regarding fragment ejection velocities and ejection angles. No theoretical development is available for the partition of explosive energy for individual fragments in

Table 13. Input and Output Files in MRSAW

X: HORIZONTAL DISTANCE(FT)
Y: VERTICAL DISTANCE(FT);
T: TIME(SEC)
XD: HORIZONTAL VELOCITY(FPS)
YD: VERTICAL VELOCITY(FPS)

FILE	DESCRIPTION
AIRBULL7.IN	INPUT FILE. PARAMETERS ARE DECRIBED IN COMMENTS
TY1EX.PLT	X VS. T FROM LINEAR ANALYSIS
TY2EX.PLT	Y VS. T FROM LINEAR ANALYSIS
Y1Y1DEX.PLT	X VS. XD FROM LINEAR ANALYSIS
Y1Y2DEX.PLT	X VS. YD FROM LINEAR ANALYSIS
Y1Y2EX.PLT	X VS. Y FROM LINEAR ANALYSIS
Y1Y2.PLT	X VS. Y FROM NONLINEAR ANALYSIS
Y1Y1D.PLT	X VS. XD FROM NONLINEAR ANALYSIS
Y1Y2D.PLT	X VS. YD FROM NONLINEAR ANALYSIS
BULLRICO.IN	INPUT FILE FOR RICOCHETING MODE FOR BULLRICO.FOR
BXY1.PLT	OUTPUT FROM BULLRICO CODE: X VS. T (FRONT TIP)
BXY2.PLT	OUTPUT FROM BULLRICO CODE: Y VS. T (FRONT TIP)
BXYG.PLT	OUTPUT FROM BULLRICO CODE: X VS. T (CENTER OF MASS)
AIRBULL7.DBG	OUTPUT FROM BULLRICO CODE: DEBUG FILE
FTIP.PLT	FINAL OUTPUT: X,Y,Z LOCI OF FRONT TIP
RTIP.PLT	FINAL OUTPUT: X,Y,Z LOCI OF REAR TIP
CMG.PLT	FINAL OUTPUT: X,Y,Z LOCI OF CENTER OF MASS G
XYZALL.PLT	FINAL OUTPUT: X,Y,Z LOCI OF FRONT, REAR TIP AND G
TXYZ12.PLT	FINAL OUTPUT: T,X,Y,Z LOCI OF FRONT AND REAR TIP
TY1.PLT	FINAL OUTPUT: T,X,Y,Z LOCI OF FRONT TIP
TY2.PLT	FINAL OUTPUT: T,X,Y,Z LOCI OF REAR TIP
Y1Y1D.PLT	FINAL OUTPUT: X,XD OF FRONT TIP
Y1Y2D.PLT	FINAL OUTPUT: X,YD OF FRONT TIP
Y1Y2.PLT	FINAL OUTPUT: X,Y OF FRONT TIP

an exploding munition and is best known empirically from test data. Due to these problems, we concentrated on describing the probability that a given area in space will be impacted by fragments of sizes within some range, and the joint probability that these fragments will have some critical energy or striking velocity to cause damage to physical objects (humans, building etc.) occupying this area. The database used in this

approach is existing arena test data published by the DoD in Joint Munitions Effectiveness Manuals (JMEM).

2.7.7 Threshold Velocities for Given target/projectile pair

For specified target-projectile combinations, it remains to establish critical impact velocities, v_{cr}, for target penetration. We also need to validate the current formulation for v_{cr}, i.e., $v_{cr} = \sqrt{\frac{2F_r d}{m}}$. This will be established from experimental data developed in Phase II. The Phase II database will be incorporated in the Phase II software for accurate determination of this mode of motion (rebound or penetration) for the second phase of the exterior ballistics. This database should encompass all projectile-targets pairs of interest. Currently, MRSAW requires user input for the determination of the mode of the second phase motion both for bullets and slugs. In the second phase of the code development this will be automated.

2.7.8 Coefficient of Restitution

In the case of a projectile rebounding from the target after impact, subsequent motion of the projectile depends on the impulsive force and its direction imparted on the projectile during impact. The time integral of this impulsive force over the time of impact is the impulse which is equal to the change in linear momentum (product of mass and velocity). This is a vector equation that relates changes in components of momentum when the impulsive force over time of impact is exactly known. However, for most problems involving impact, this information is not easy to obtain even from very controlled experiments. Since frictional forces (or lateral force) is usually small compared to the force exerted to the projectile in the direction normal to the target surface at the point of impact, the momentum equation gives a conservation of momentum in the direction tangential to the surface of the target. To calculate the speed and direction of the projectile after impact, we use the classical law of impact which states that the velocity of separation after impact is equal to the product of a constant e (called the coefficient of restitution) and the velocity of approach prior to impact. Experiments on impacting bodies show that e depends on the projectile-target pair. In the development of MRSAW, we have assumed that e is known for a given projectile target pair. In Phase II we will experimentally create a database for e employing projectile-target combinations of interest. For any given projectile-target pair, e can be calculated by measuring the speed of rebound and the angle of rebound of the projectile and using the impact law and momentum equation mentioned above.

2.7.9 Determination of Spray Angle and Number of Pellets within a given radius r at a given range R from Experimental Data

The mathematical formula that predicts the number of pellets $n = n(R,r)$ within a certain radius r at range R for a given shot in a burst mode, has the form:

$$n(R,r) = N.e^{-\alpha\left(\frac{R}{R_f}\right)\left(\beta-\frac{r}{R}\right)} \qquad (15)$$

where α and β are model parameters, R_f is the reference range and N is the total number of pellets in the shot.

This formula is consistent with the following physical conditions that need to be satisfied by the function $n(R,r)$:

1. At very close range, R approaches zero and $n(R,r)$ should approach N
2. At a given range R, $n(R,r)$ should be an increasing function of the radial variable, r.
3. r should lie between 0 and its maximum $R \tan\theta$ where θ is the spray angle.
4. For any given range R, $n(R,R \tan\theta)=N$. This gives $\beta=\tan\theta$

Equation 15 for $n(R,r)$ satisfies all these conditions, and hence is a candidate function for predicting the number of pellets at any given range, R, and within any given radius, r. Its applicability depends on its ability to predict experimental results and predicted trends should be physically plausible.

To apply the above results for any given shot with known number of pellets N, we need to have at least two experimental results where the number of pellets n within a given radius, r, at a given range, R, are known. Let these two data points be: (1) n=n1, R=R1, r=r1; and, (2) n=n2, R=R2, r=r2. (The experimental results below were obtained from the ammunition manufacturer and were reported in the Sixth progress report for this effort).

First Experiment

Shot Size 1: Pellet Count = 16: Muzzle Velocity = 1213 fps
Energy per pellet 127.8 ft.-lbs

Experimental Result: At 40 yards range,
4 pellets within a radius of 20 in.
9 pellets within a radius of 30 in.

Thus, n1=4, r1=20, R1=40x3x12 and n2=9, r2=30, R2=R1

N := 16
n1 := 4
r1 := 20
R1 := 40·3·12
n2 := 9

$r2 := 30$

$R2 := R1$

$Rf := 40 \cdot 3 \cdot 12$

$Rf = 1.44 \cdot 10^3$

$$\beta := \frac{\frac{r1}{R1} \cdot \ln\left(\frac{n2}{N}\right) - \frac{r2}{R2} \cdot \ln\left(\frac{n1}{N}\right)}{\ln\left(\frac{n2}{n1}\right)}$$

$\beta = 0.026$

$\theta := \operatorname{atan}(\beta)$

$\theta = 0.026$

$$\alpha := \frac{\ln\left(\frac{N}{n1}\right)}{\left(\frac{R1}{Rf}\right) \cdot \left(\beta - \frac{r1}{R1}\right)}$$

$\alpha = 116.774$

$$n(R, r) := N \cdot \exp\left[-\alpha \cdot \frac{R}{Rf} \cdot \left(\beta - \frac{r}{R}\right)\right]$$

Check Results: Theoretical Prediction

Check Data:

$R := 40 \cdot 3 \cdot 12$

$R \cdot \beta = 37.095$

$r := 20$

$n(R, r) = 4$

$r := 30$

$n(R, r) = 9$

OK

Predictions:

$R := 50 \cdot 3 \cdot 12$

$R \cdot \beta = 46.369$

$r := 20$

$n(R, r) = 1.886$

$r := 30$

$n(R, r) = 4.243$

$r := 40$

$n(R, r) = 9.546$

Second Experiment:

Shot Size 4: Pellet Count 27: Muzzle Velocity 1268 fps
Energy per pellet 67.3 ft.-lbs

Experimental Result: At 40 yd. range,
9 pellets within a radius of 20 in.
15 pellets within a radius of 30 in.

$N1 := 9$

$N := 27$

$r1 := 20$

$R := 40 \cdot 3 \cdot 12$

$N2 := 15$

$R = 1.44 \cdot 10^3$

$r2 := 30$

$Rf := 40 \cdot 3 \cdot 12$

$Rf = 1.44 \cdot 10^3$

$$\beta := \frac{\frac{r1}{R} \cdot \ln\left(\frac{N2}{N}\right) - \frac{r2}{R} \cdot \ln\left(\frac{N1}{N}\right)}{\ln\left(\frac{N2}{N1}\right)}$$

$\beta = 0.029$

$\theta := \operatorname{atan}(\beta)$

$\theta = 0.029$

$$\alpha := \frac{\ln\left(\frac{N}{N1}\right)}{\left(\frac{R}{Rf}\right) \cdot \left(\beta - \frac{r1}{R}\right)}$$

$\alpha = 73.559$

$$n(R, r) := N \cdot \exp\left[-\alpha \cdot \frac{R}{Rf} \cdot \left(\beta - \frac{r}{R}\right)\right]$$

Check Results: Theoretical Predictions

$R := 40 \cdot 3 \cdot 12$
$R \cdot \beta = 41.507$
$r := 20$
$n(R, r) = 9$
$r := 30$
$n(R, r) = 15$ OK

Other Predictions: At a range of 50 yards,

$R := 50 \cdot 3 \cdot 12$
$R \cdot \beta = 51.883$
$r := 20$
$n(R, r) = 5.297$
$r := 30$
$n(R, r) = 8.828$
$r := 40$
$n(R, r) = 14.714$
$r := 45$
$n(R, r) = 18.996$
$R := 5 \cdot 12$
$R = 60$
$R \cdot \beta = 1.729$
$r := 1.5$
$n(R, r) = 26.685$

Plot of Pellet Distribution as a Function of r for any given range R

The functional form of the distribution of pellet within a circle of radius r at range R is given by:

$$n(R,r) = N.e^{-\alpha\left(\frac{R}{R_f}\right)\left(\beta-\frac{r}{R}\right)}$$

where α is a model parameter and $\beta=\tan\theta$ where θ is the spray angle. Rf is the range where experimental data is collected. In this case Rf=40 yards=1440 inches.

For the first experiment we have $\alpha=116.774$, $\beta=0.026$, N=16

$\alpha := 116.774$

$\beta := 0.026$
$R := 40 \cdot 3 \cdot 12$
$R = 1.44 \cdot 10^3$
$N := 16$
$Rf = 1.44 \cdot 10^3$
$R \cdot \beta = 37.44$
$r := 0.75, 1.50 .. 37.44$
$n1(r) := N \cdot \exp\left[-\alpha \cdot \frac{R}{Rf} \cdot \left(\beta - \frac{r}{R} \right) \right]$

For the second experiment we have α=73.559, β=0.029, N=27

$\alpha := 73.559$
$\beta := 0.029$
$R := 40 \cdot 3 \cdot 12$
$R = 1.44 \cdot 10^3$
$N := 27$
$Rf = 1.44 \cdot 10^3$
$R \cdot \beta = 41.76$
$r := 0.72, 1.44 .. 41.76$
$n2(r) := N \cdot \exp\left[-\alpha \cdot \frac{R}{Rf} \cdot \left(\beta - \frac{r}{R} \right) \right]$

Graphical Representation of Target Damage due to:

1. Burst Mode Shot (3 rounds)
2. Automatic Mode Shots
3. Shots with Circular Error Probability (CEP)

2.7.10 Burst Mode Shots

In this case, the distribution of number pellets $n(R,r)$ inside a circular region of radius r at any given range R, is given by the function:

$$n(R,r) = N.e^{-\alpha\left(\frac{R}{R_f}\right)\left(\beta-\frac{r}{R}\right)}$$

For any given shot, the model parameters α *and* β can be calculated from experimental data on the number of pellets for at least two pairs of values of R and r.

The methodology of calculating these model parameters have been explained in the previous sections. Recall that β is related to the spray angle θ through $\beta = \tan\theta$ and *N* is the total number of pellets in the shot.

For the graphical representation of damage area on a plane target located at a range *R* from the shot location, we first calculate the radius of coverage r_{max} from $r_{max} = R\tan\theta$. Since this area is hit by *N* pellets at random locations, we first generate *N* random numbers within *0* and *1* using a random number generator. The *i-th* number (i=1,2,...,N) is scaled such that it lies between r_{i-1} and r_i (r_o=0) where $r_i = R\left[\beta - \ln\left(\frac{N}{i}\right)\left(\frac{R_f}{R}\right)\frac{1}{\alpha}\right]$. This is the inversion of Equation (15) and R_f is the range used in the experimental determination of α and β from Equation (15). These numbers give the radial coordinates of the shot points inside the target. To determine the circumferential locations for these shots, we generate another set of *N* random numbers between *0* and 2π. These give the angular coordinates of the shots on the target. These results are then shown graphically.

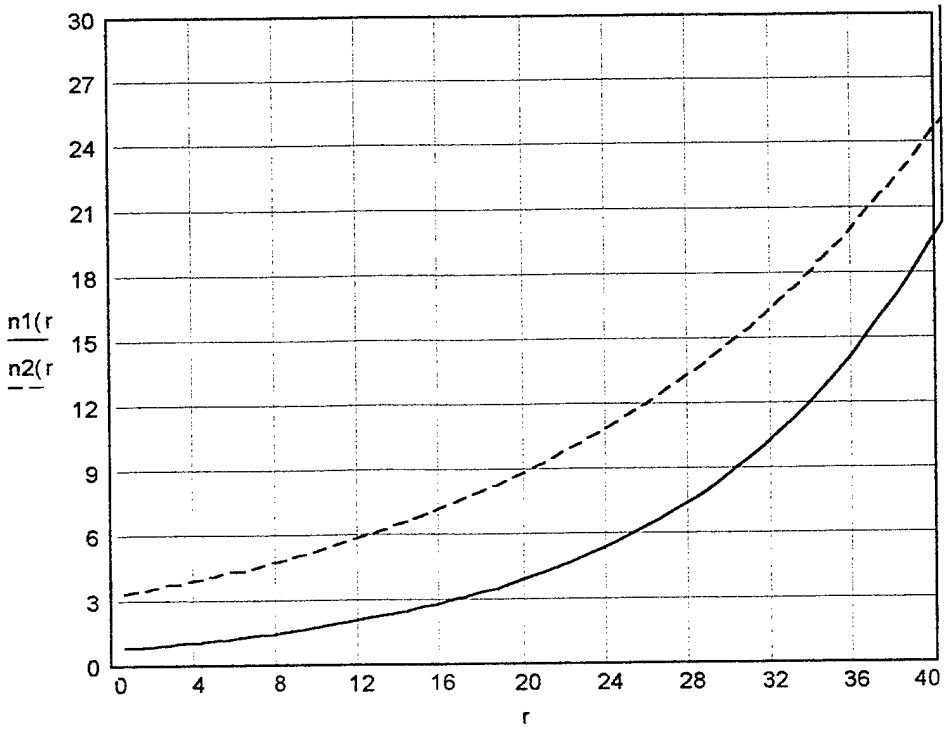


Figure 12. Number of Pellets versus Radius at a Range of 40 yards

2.7.11 Automatic Mode

In the case of automatic mode, we assume that the muzzle velocities remain the same and only the shot locations may vary within certain range while the shot angles varies between some maximum and minimum angles with the horizontal. In this case, the shots on target should lie within a rectangular region where the vertical boundaries can be established from the maximum and minimum shot angles from locations lying on the vertical boundaries on the shot plane using the analytical code MRSAW. It may be recalled here that for the determination of the shot location on a given target, MRSAW requires the muzzle velocity, shot location and shot angle with the horizontal. Horizontal boundaries will be identical to the horizontal boundaries within which the automatic shots are fired. Once the boundaries of the rectangular shot area on the target are determined, graphical representation of the damage area on the target requires the generation of N (N is the number of shots in the automatic mode) pairs of random number generations which are scaled to lie within the rectangular boundary. This methodology requires only two runs of the MRSAW code instead of the N number of shots actually fired from the shot plane.

2.7.12 Soldier Stress and Aiming Errors

In this case, the damage region on a target at a given range is influenced by the person firing the shots. The dimensions describing the damage depends on whether the person firing these shots are under stress or not. The distribution functions describing the linear dimensions of the damage region at a given range are available in the literature for both stressed and unstressed people. For the graphical representation of the damage zone, we need to generate *N* (number of shots fired) pairs of random numbers lying with the boundary dictated by the known distribution functions at a given range. MRSAW has been programmed to read this data.

3. VISUALIZATION AND SCENE ANIMATION

The visualization and scene animation task, Task 6, identified requirements, tested methodologies, and evaluated third-party software for visualization and implementation of effects in the Phase II virtual environment. Two strategies were considered for developing the Phase II virtual environment.

The first approach was to develop wrap-around code for an existing commercial visualization system. Several candidate commercial systems were considered for this application including MultiGen ™, Coryphaeus ™, and Division ™.

The second approach was to custom design, develop, and integrate a three dimensional modeling and real-time simulation system. This system would be tailored specifically for the types of objects, environments, and scenarios which would be simulated in the Phase II system. Both approaches would yield compelling simulations of the virtual environment however implementation of this second approach was quickly seen to exceed resources available in the Phase II effort.

The proposed approach is a hybrid of the above strategies and develops the virtual environment and interfaces by integrating commercially available off-the-shelf hardware and software. During Phase I, representative hardware and software was evaluated for compatibility with the effects to be implemented in the Phase II virtual environment. The effects to be implemented were flowed down from the Scenario Development task, Task 1, the Damage-Response Mode Matrix task, Task 2, the Weapons Effects task, Task 3, and the Structural Integrity task, Task 4.

3.1 SOFTWARE EVALUATION

Existing commercial software systems were evaluated to understand constraints associated with the real-time system development in Phase II and to gain insight as to how current state-of-the-art technology in real time modeling and simulation could be exploited for the Phase II effort. This would enable us to determine whether off-the-shelf software could be used for this project and also what deficiencies, if any, exist in the software. While many real time modeling and simulation packages are available, we focused our evaluation on two competing packages, MultiGen Inc. (San Jose, CA) and Coryphaeus Software Inc. (Los Gatos, CA). Both systems were chosen because of their widespread use within the Department of Defense.

The main criteria used to determine the effectiveness of the software (in no particular order) were:

- modeling features
- texture mapping

- dynamic lighting capabilities
- special effects (explosions, smoke, fire, debris, flak, etc.)
- animation features
- audio capabilities
- DIS compatibility
- ease of learning and ease of use
- programmability and customization
- support for other data formats
- real time data from external application
- portability

3.1.1 MultiGen™

The first tool evaluated was MultiGen. While Multigen proved to be a very strong 3D modeling package for real-time simulations and satisfied many of the established criteria. The special effects available as options in the software were limited for our application. Once 3D models were completed, there was no easy way to view them in real-time due to a lack of a simulation development environment. MultiGen is primarily a real time 3D modeling tool and not a real time simulation development tool. It was determined that we needed to evaluate real time simulation development environments in addition to 3D modeling environments.

3.1.2 Coryphaeus ™

Designer's Workbench™ (3D modeling package) and EasyScene™ (real time development environment) from Coryphaeus Software Inc. were evaluated. *Designer's Workbench* proved to be a good modeling package for real-time databases. While somewhat limited in complex geometry creation tools, it has sufficient tools for creating the simple geometry required for real-time applications. Also, the ability to read geometry from a variety of other modeling packages and other real-time applications helps make up for modeling deficiencies.

Texture mapping is straight forward using the graphical user interface. Unfortunately, even a texture mapped cube requires more graphics power than is available on a SGI Indigo2 Extreme™ which we were using during the Phase I activity. This proved to be the limiting factor in evaluating texture mapping performance.

Most of the lighting options are geared more toward aircraft simulations but it is possible to simulate lighting conditions inside buildings as well. To add depth to the simulation, shadows can be activated. Infrared lighting conditions can also be simulated without much difficulty.

As with the other real time 3D packages tested, Coryphaeus has a limited selection of special effects. However, additional special effects modules can be written in C/C++ and incorporated into the simulation. Since our application requires debris and ejecta

for a wide variety of materials and munitions, "canned" special effects would not be utilized exclusively anyway.

Audio effects can be attached to objects or events and triggered within a simulation. Thus, it will be possible to record the sounds from actual explosions and play them during the triggered event. This will enhance the perceived realism of the experience.

Coryphaeus claims to have built in DIS capability to allow access to a DIS network without writing additional code. DIS/HLA requirements are undergoing massive revisions however and these issues will have to be re-examined during Phase II as new requirements unfold and as the wrap-around software is being developed.

Programming additional features and creating user defined modules will be a very key part of the phase II effort. Coryphaeus has hundreds of user functions which can be called from a C/C++ program and incorporated into the simulation. Coryphaeus is also totally compatible with IRIS Performer ™ (Silicon Graphics ANSI C and C++ programming interface for creating real-time visual simulation and other interactive graphics applications) which offers additional functions and features. There is also support for multiple channels of input/output data, multiple CPU's, and multiple graphics pipes (to feed graphics to multiple display devices). These features will enable optimization of the hardware and software architecture environment for real time performance.

One very strong feature of Coryphaeus is the ability to import data from other formats and applications. This enables us to take advantage of the wide variety of 3D geometry which have been created for other applications. Geometry can be imported from about 30 other modeling and real-time applications including Wavefront ™ (advanced visualization and inverse kinematics software commonly used in scientific visualization), Alias ™ (advanced surfacing and animation software commonly used in design and entertainment), IRIS Performer ™ (Silicon Graphics ANSI C and C++ programming interface for creating real-time visual simulation and other interactive graphics applications), Lightscape ™. (A visualization system combining proprietary radiosity algorithms with a physically based lighting interface commonly used in the architecture industry). The extensive array of 3D models available from Image data can be imported for use as textures and special effects. Motion and event data can also be imported and applied to objects in the database. This will be crucial as the algorithms which describe weapons effects will be incorporated in this way. The only computer platform Coryphaeus software currently runs on is Silicon Graphics but this is a limitation shared by many other visualization tools as well.

3.2 DEVELOPMENT APPROACH

Several *off-the-shelf* solutions exist for creation of 3D geometry for real time simulations. Both Coryphaeus and Multigen were capable of creating the objects

Figure 13. Ceiling Impact Surface following Ballistic Test with 12-guage shotgun using Number 4 buckshot from 8 feet.

necessary for MOUT type virtual environments. It was discovered however that the modeling activity is a less significant element of the overall simulation than initially perceived. That is, development of the weapons effects simulation is much more complex and represent more stressing constraints on Phase II development.

The software evaluated contained tools for creating object trails (aircraft simulations), weather conditions (time of day, fog, visibility, background color), and triggerable effects (explosions, flak, and debris). Explosions are represented as reddish orange plumes of flames and smoke that dissipate from a model. Flak is represented as a light gray and black plume that dissipates from the model. Debris is viewed as small particles that move outward from the origin of the model. While these triggerable effects are adequate when viewed from the far-field (from an airplane to the ground for example), they are not as useful for near field views of weapon effects prevalent in an urban warfare application (refer to Figure 13 and Figure 14).

To enhance the realism of the effects, additional programming will be necessary. We can build on the base effects and add the functionality necessary to create convincing simulations of our weapons effects. This will be accomplished by developing ANSI C and C++ routines which call functions from Coryphaeus API, IRIS Performer, X, and OpenGL ™ (platform independent graphics libraries and functions developed by SGI) executable libraries. These routines are then compiled and attached to the simulation at run-time.

In conclusion, a hybrid approach that uses off-the-shelf commercial software in combination with custom developed software will be necessary. The Coryphaeus products evaluated will form the foundation for the development environment. Most, if not all, of the 3D environment will be created using Designer's Workbench. EasyScene will be used to create the basic simulation environment. Additional programming will be

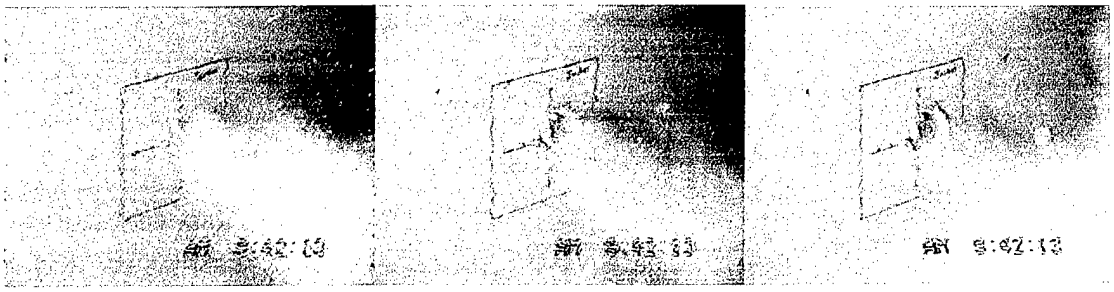


Figure 14. Backface Debris and Ejecta from Interior Sheet Rock Wall subject to 12 gauge shotgun with #4 Buckshot fired from 8 feet

(STRICOM Phase I Ballisitc Experiments conducted by MRC)

implemented to develop the weapons effects required by urban warfare and the methodology necessary to link the weapons effects algorithms with the weapons effects visuals.

3.3 McKenna MOUT SITE DEVELOPMENT

For purpose of software evaluation, the McKenna MOUT Town Hall building (shown in Figure 15) was created using Designer's Workbench from Coryphaeus. The geometry accurately represents the engineering drawings as supplied by the United States Army Infantry School at Fort Benning. All walls, windows, and doors on all three floors have been accurately positioned and the stairwell in the center of the building allows access to all floors. The floor plan for the ground floor is illustrated in Figure 16. Figure 17

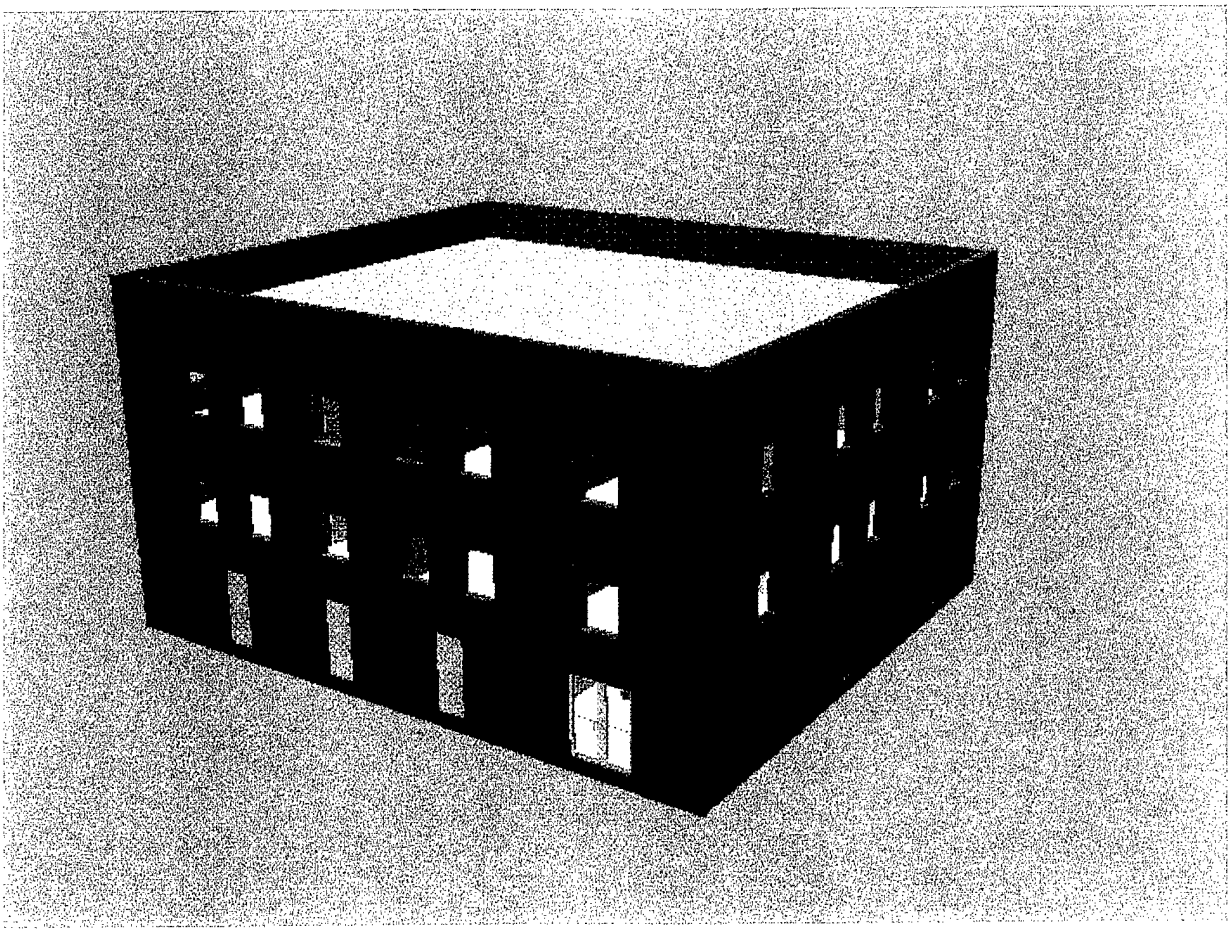


Figure 15. Exterior of McKenna MOUT Site Town Hall Building

illustrates the main entrance of the McKenna MOUT Town Hall Building, and Figure 18 shows the stairwell.

With some effort, the entire building can be explored in near real-time. A simple fly-through of the McKenna MOUT Town Hall Building was provided to STRICOM in VHS

format. The original intent, as is the case with all real-time efforts, was to utilize texture mapping as much as possible to minimize the need for geometry. However, the graphics requirements of texture mapping exceeded the performance capabilities of the SGI Indigo^2 Extreme used during the Phase I effort. Even a cube with a brick texture map rendered the machine nearly useless for development. The cube rotated very

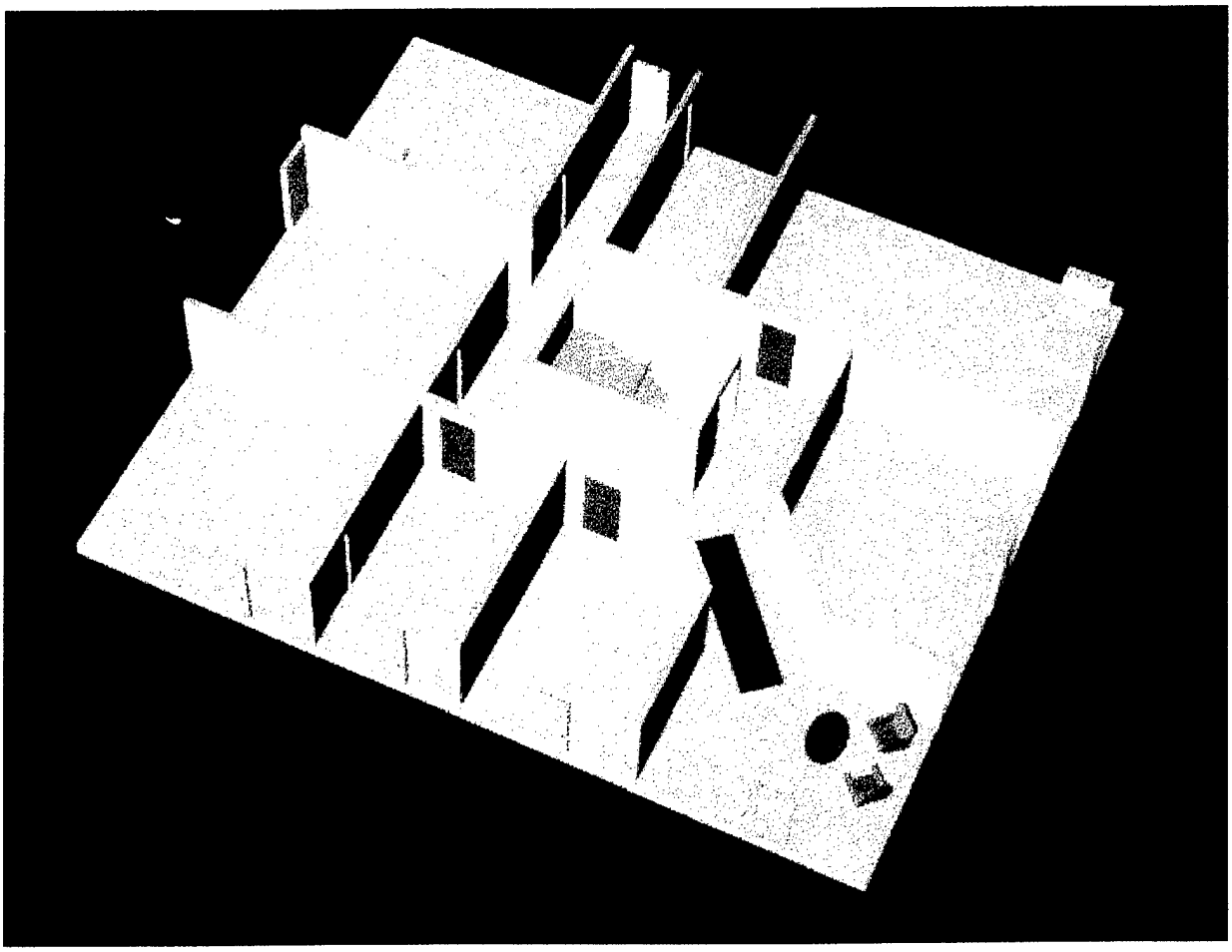


Figure 16. First Floor McKenna Town Hall Layout

slowly and was jerky and all menus responded several seconds after selection. Thus, more geometry was created and simple materials were used to represent surfaces. No textures were used. This is only a temporary problem as any implementation of the final system would undoubtedly run on a high performance computer system (SGI High Impact or better). When interior photos are available and satisfactory hardware is available, a very realistic representation of the McKenna MOUT site is achievable. The Phase I building is meant to be used as a representative MOUT object not an exact replica.

To conserve resources during Phase I, several issues necessary for a compelling weapons effects simulation were investigated decoupled from the algorithms describing physical effects and the Town Hall fly-through developed in this and other tasks of the Phase I effort. That is, the weapons effects simulations were evaluated for availability and effectiveness but not developed to the point where the simulations could be integrated into the Town Hall fly-through. This was done to highlight requirements that need to be imposed on the Phase II virtual environment and to allow adequate time to

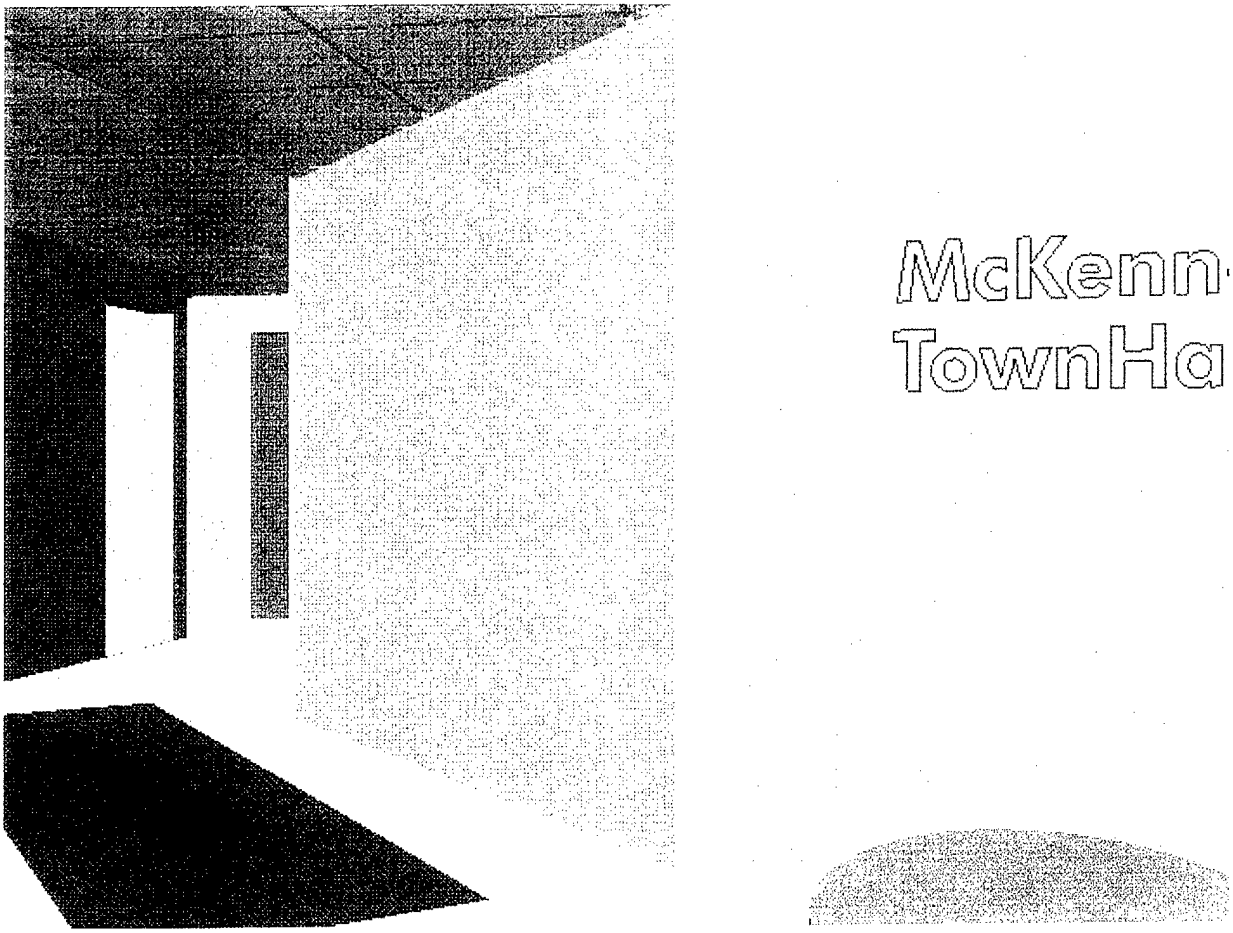


Figure 17. Town Hall Main Entrance

evaluate off-the-shelf software in meeting these requirements. Dynamic lighting, animation effects (opening and closing doors, etc.), collision detection, and level of detail were evaluated in this manner. For example, a simple model of a door and a wall was created to investigate the mechanism of animating the opening and closing of the door. Effects or functions requiring texture mapping (smoke, dust, fire, fabric, wall hangings, etc.) were not evaluated due to the hardware limitation previously mentioned.

3.4 Weapons Effects Visualization

The primary elements of convincing weapons effects in an immersed environment are:

- sound effects (of weapon fire and impact with objects)
- dust, flak (airborne particulate created at bullet entry and exit from object)
- debris (pieces of objects remaining after impact either around penetration hole or on the floor below the hole)
- smoke and fire (when appropriate)
- visible change of geometry (actual holes in objects or surfaces

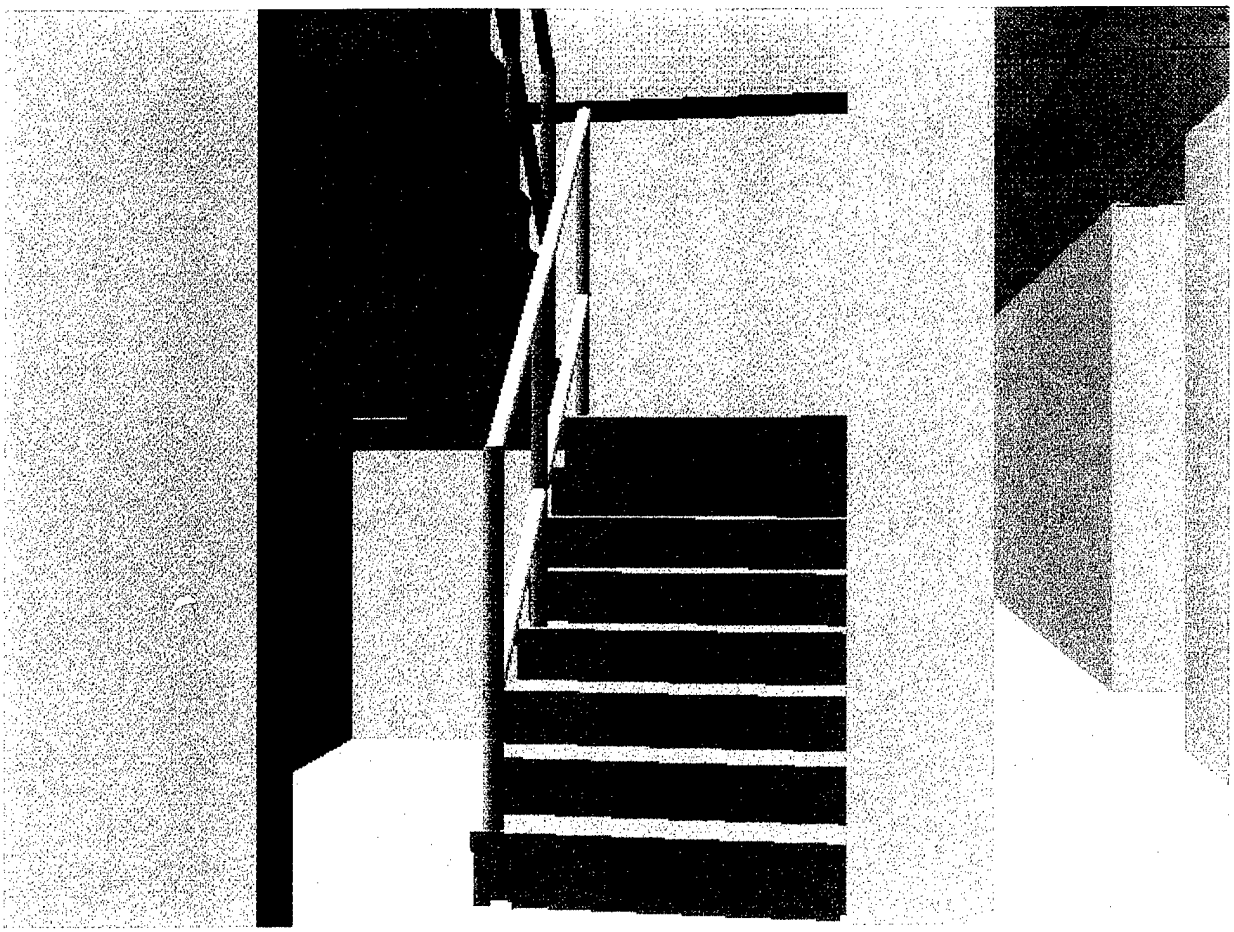


Figure 18. Town Hall Stairwell

If any of these factors are missing, the simulation will be less effective. Sound effects of weapons triggered at the appropriate time are not a problem. Audio recordings from ballistic experiments will be adequate for first level realism. However, all other factors indicated above will require additional development outside of the Coryphaeus environment. While some of the necessary controls are available for dust, smoke, and fire, they were not considered sufficiently robust to handle the variety of munitions and material combinations possible in the MOUT environment.

The debris resulting from an explosion or from a weapon being fired into a surface, as well as the visible change in the affected surface geometry present the biggest challenge. What level of realism is necessary? First level realism can be achieved by digitally pasting a photo of an actual bullet hole onto the surface where the bullet penetrated (Figure 19).

The photo (texture) would come from a library of digitized photos obtained from ballistic experiments of various materials and munitions. The weapons effect would be completed by adding the effect of dust, debris, or ejecta resulting from the bullet.

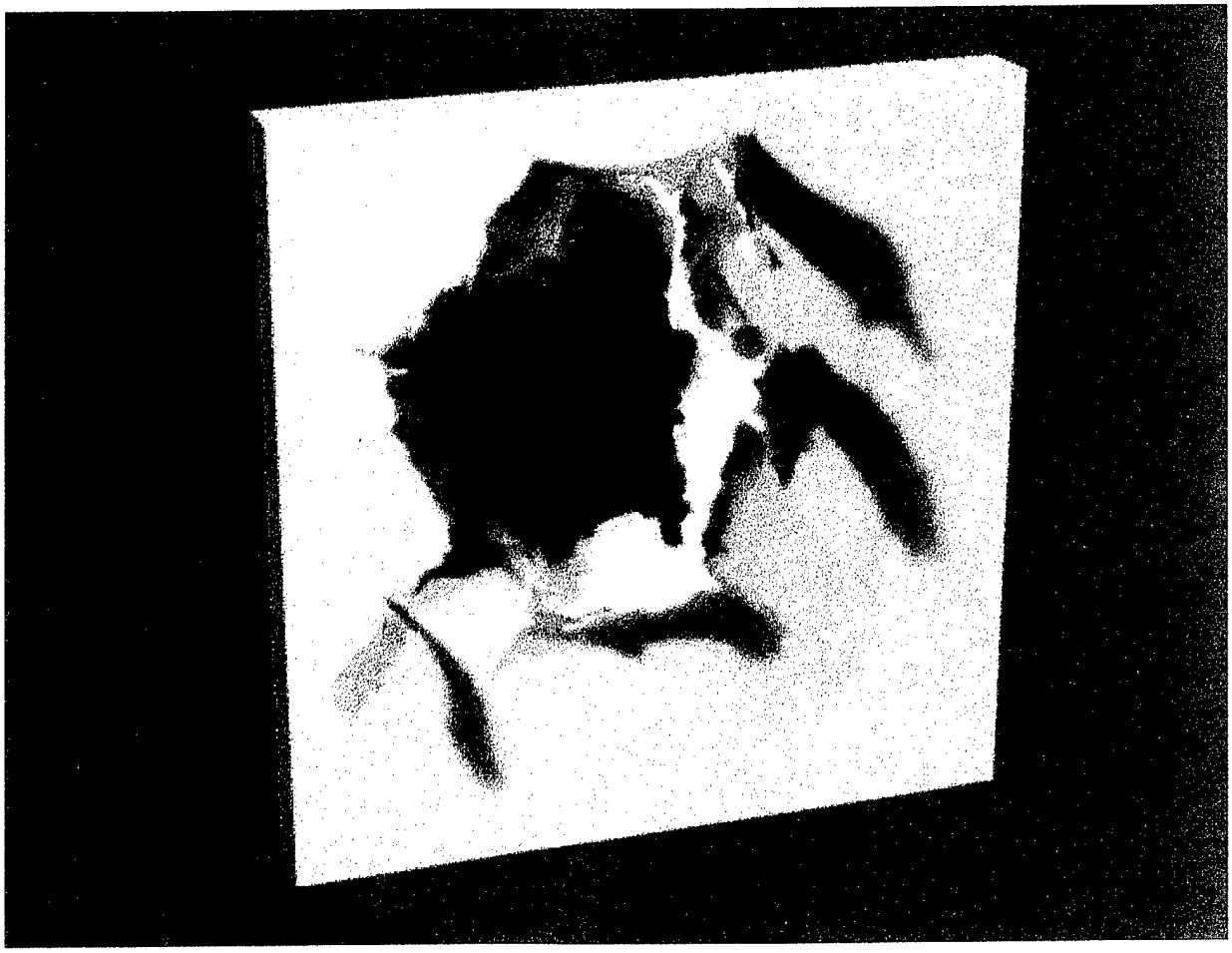


Figure 19. Texture Mapped Surface of Impact Surface of Interior Wall with 12 gauge Shot Gun Hole (Number 4 buckshot and range of 8 feet).

A problem with this approach is that it can't account for the situation where sizable holes are created by the weapon. For example, if a hole large enough to look through into the next room is strategically placed in a wall, a texture map cannot alone be used for the simulation. Applying the texture of a hole to a surface does not create a hole in

the surface. In that way, a texture map can be thought of as wallpaper. If the wallpaper does not have a physical hole, placing it on a surface will not create a hole.

The only way to simulate this situation and achieve a higher level of realism is by geometric modification. If we knew where to hole was to be located each time the simulation was entered, we could create a hole in the surface in advance and "uncover" it at the appropriate time. Since a weapon can be fired at any surface in the simulated environment from any direction, the possible locations for the "hole" are innumerable. Thus, we must develop a method to reconfigure the geometry of the affected surface in real time.

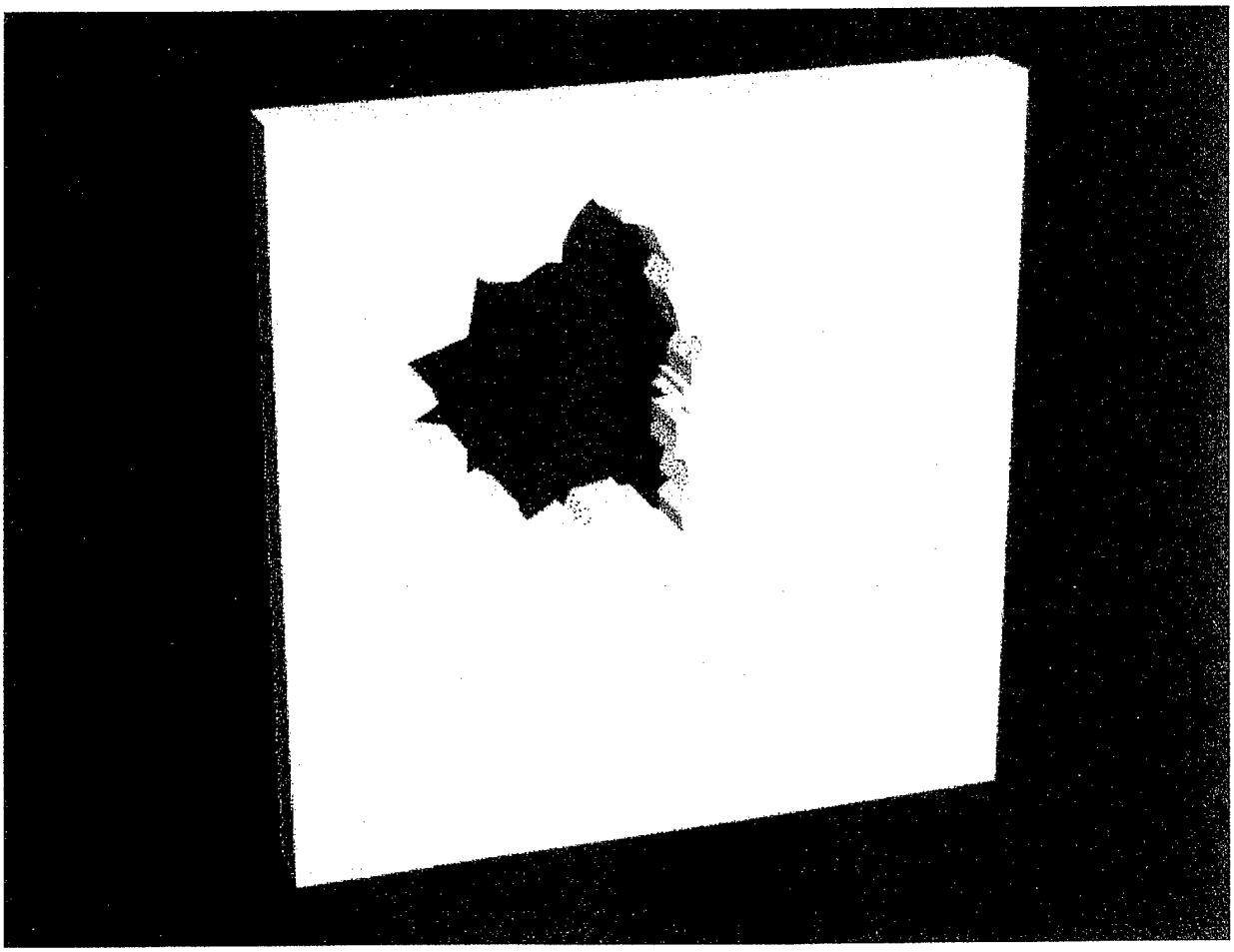


Figure 20. Interior Wall with Hole from 12 gauge shotgun and Number 4 Buckshot

Adding the texture (photo) of the hole will provide the additional detail necessary for the debris around the hole (refer to Figure 21). The addition of an effect to simulate dust, debris, or ejecta from the penetration will complete this effect.

The weapons effects algorithms developed in Tasks 3 and 4 can already predict the hole pattern resulting from various weapons and material combinations. Also, data is available from the simulation to precisely locate in 3-dimensional space where the weapon was fired from and the coordinates of each object in the environment. With this information available, it is possible to develop a function which will take the geometry of a wall for example, and re-create the wall to contain the hole created by the weapon. This new wall geometry would then be passed back to the simulation where it will replace the original wall until the simulation is re-initialized.

The wall with the hole (see to Figure 20) has been created by digitally "tracing" the hole in the photo and removing that portion of the wall. It remains to be determined what level of the geometric reconfiguration calculations can be performed in real time. As the

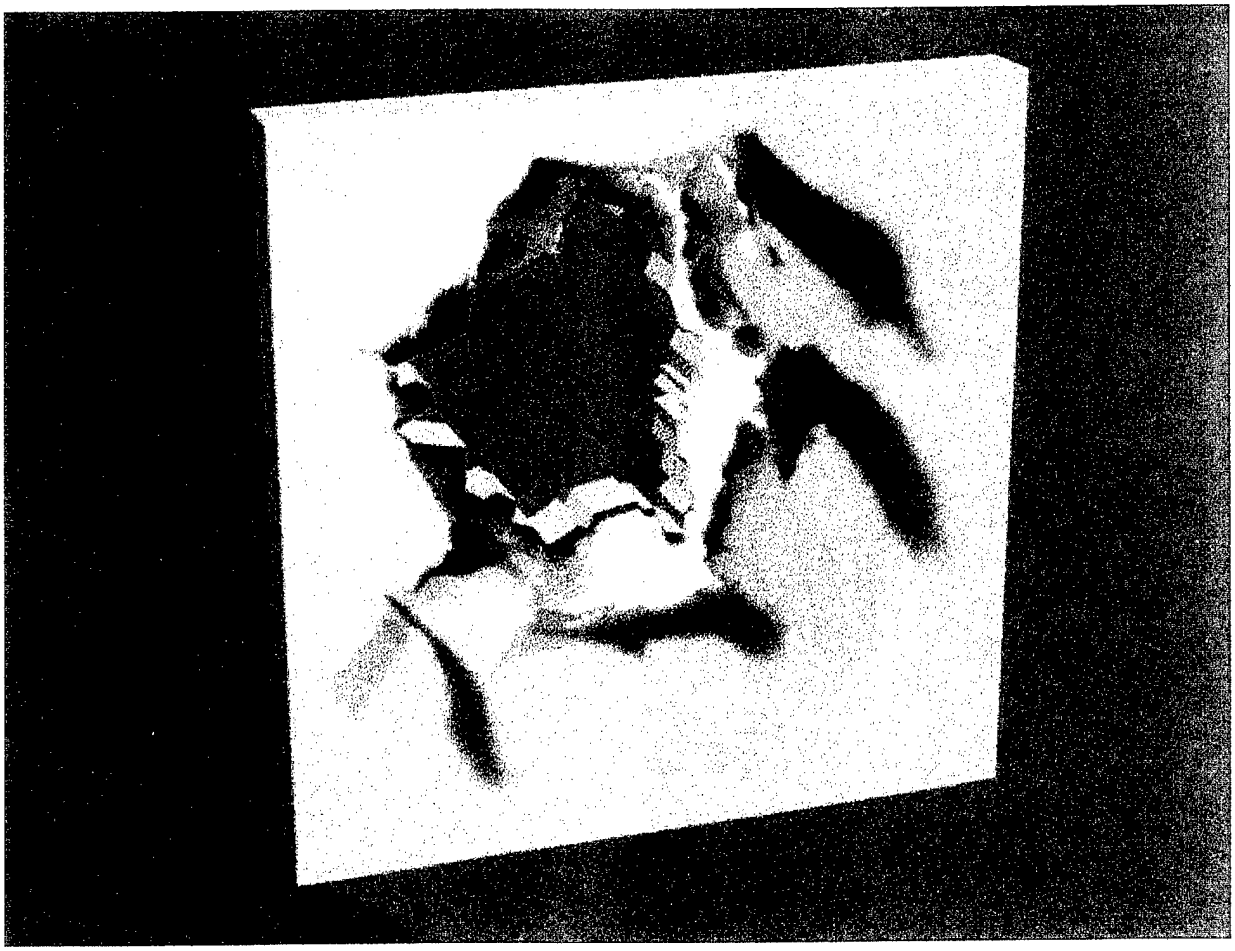


Figure 21. Object with Geometric Hole and Applied Texture Map

simulation progresses, more and more weapons will be fired, resulting in more weapons effects being simulated. This creates a potential problem for real time geometric reconfiguration calculations and real time display of the resulting 3D

environment. Figure 22, Figure 23, and Figure 24 illustrates the accumulative polygonal effect created by adding more and more bullet holes to an object. Figure 22 is the polygonal representation of wall, Figure 23 is the polygonal representation of a wall with one bullet hole, and Figure 24 is the final wall with multiple holes.

Note that the polygon count has increased from 6 to 807 polygons. If 50 bullets holes were created instead of 5, the polygon count would be over 7000! This increase dramatically illustrates why texture mapping is an attractive option for far field viewing. Graphics hardware can display 50 texture mapped bullet holes much easier than an additional 7000 polygons. It is very important to use geometric reconfiguration cautiously. The algorithm would need to be robust enough to determine, based on viewing distance, whether a texture map should be used (Figure 19) instead of actual geometry (Figure 21).

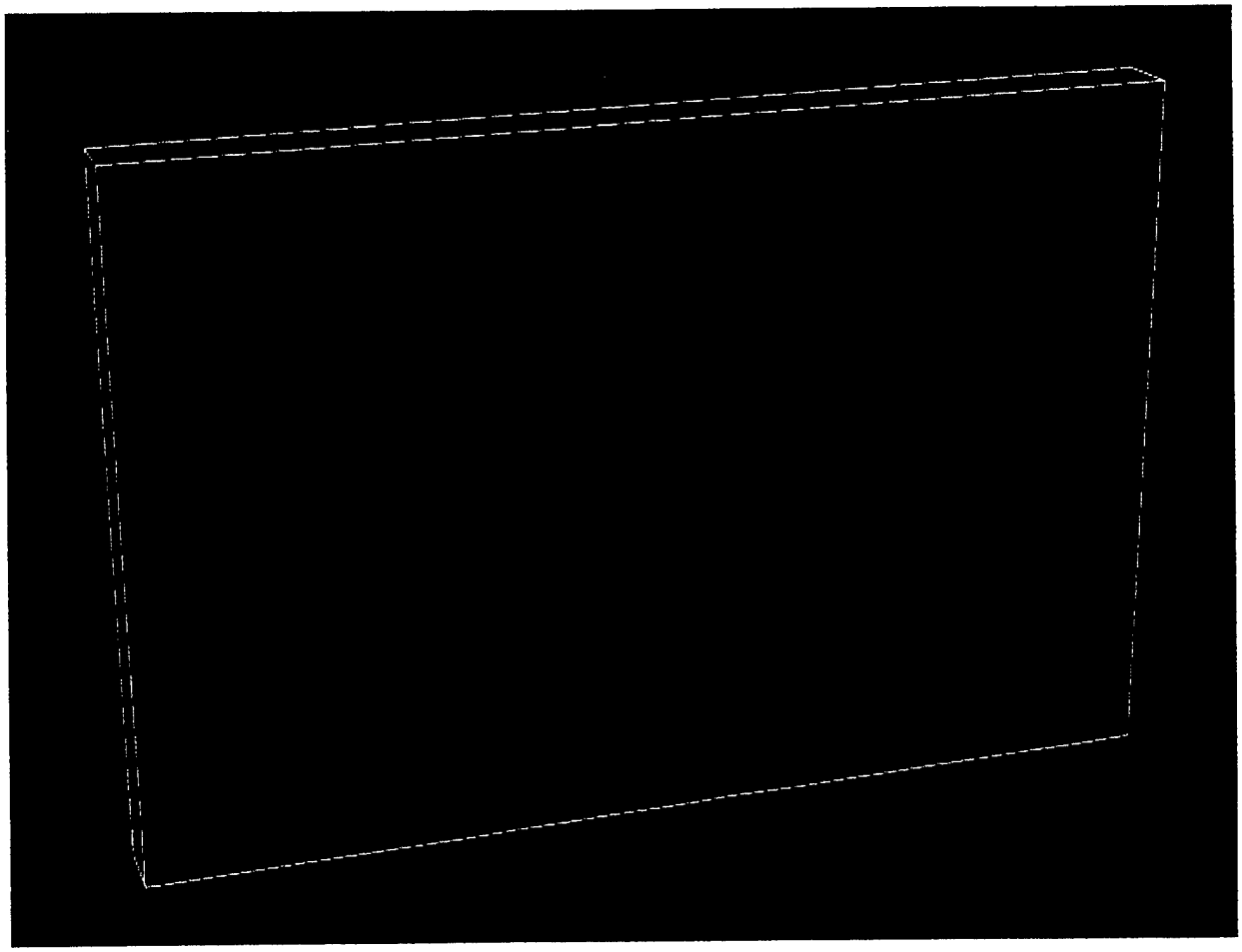


Figure 22. Solid Wall – 6 Polygons

Another option is to use simplified bullet holes. For the bullet hole in Figure 20 we "traced" the outline of the bullet hole. We could instead use a 6-10 sided circle for the hole and still create a hole in the surface. This is not as visually compelling but would

reduce the polygon count significantly. This option could be used as the viewer walks toward the wall. The actual hole could be substituted as the viewer reaches the wall.

If the initial geometry of all walls and objects in the environment is kept simple (minimum curvature, minimum surface detail) and a dedicated processor is used for the calculations (this can even be a PC based computer), satisfactory geometric reconfiguration performance can be achieved. Also, implementing level of detail control into the geometry reconfiguration algorithm (realism of bullet hole shape as a function of viewer distance from the surface) will help minimize polygon count and thus increase display performance.

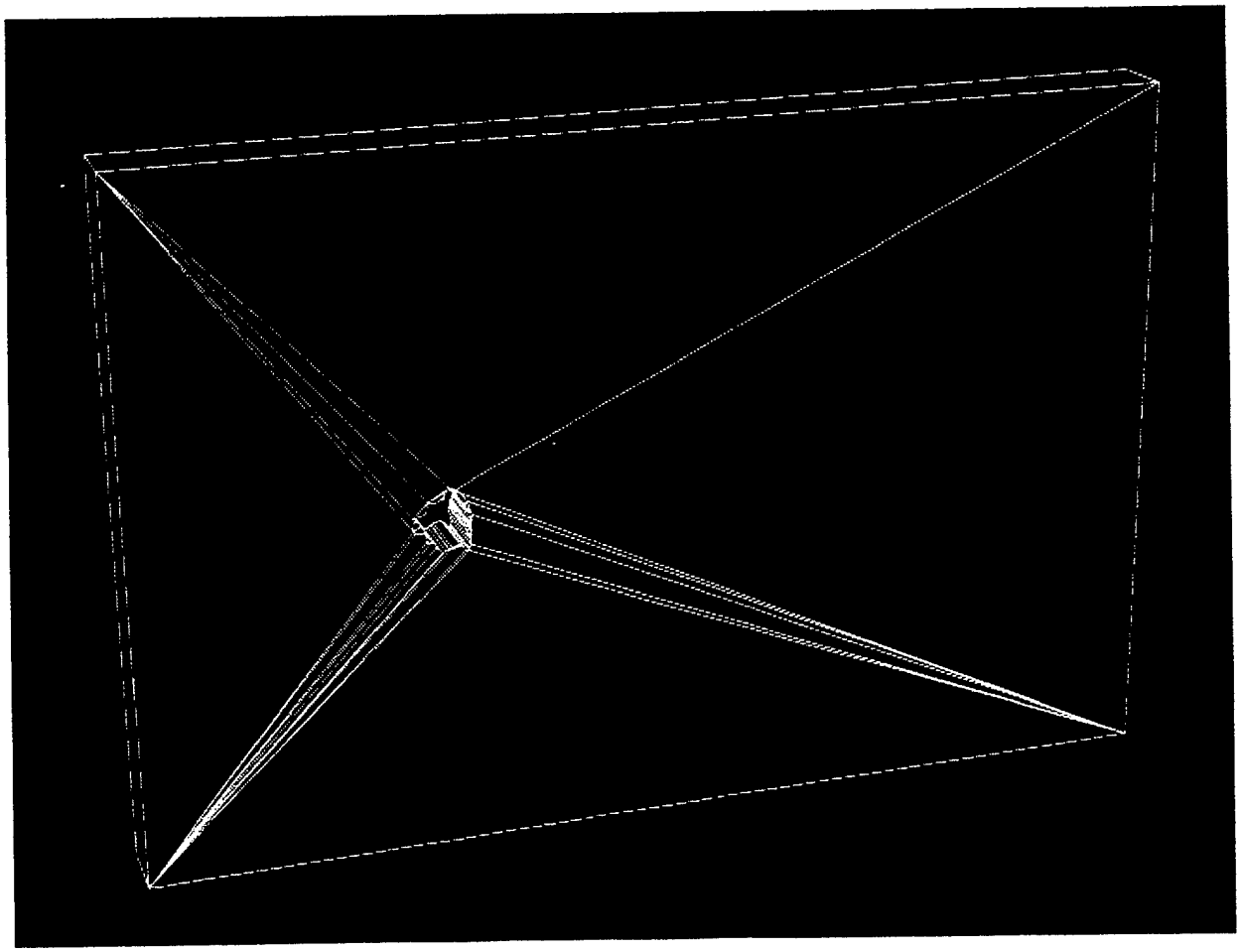


Figure 23. Wall with One Bullet Hole – 159 Polygons

3.5 Hardware and Software Architecture

The overall simulation structure can be designed to execute as a single UNIX process. However to permit real time performance, an environment with more than one processor is desirable. A high end SGI computer (SGI High Impact or better) will be

required for the graphics display of the simulation. To achieve real time performance, it is important to perform as many auxiliary computations (weapons effects algorithms, geometric reconfiguration function, etc.) as possible on separate processors. The additional processors could be Pentium ™ based PC's. As more characters are immersed in the simulation, additional graphics processors would be required.

Each immersed character, in addition to having some type of tracking device, will also have one or more weapons. At this time, no significant effort has been expended to evaluate these devices. The core system proposed in phase one has built in capabilities to handle the input from these devices and any specific issues will be revisited during Phase II.

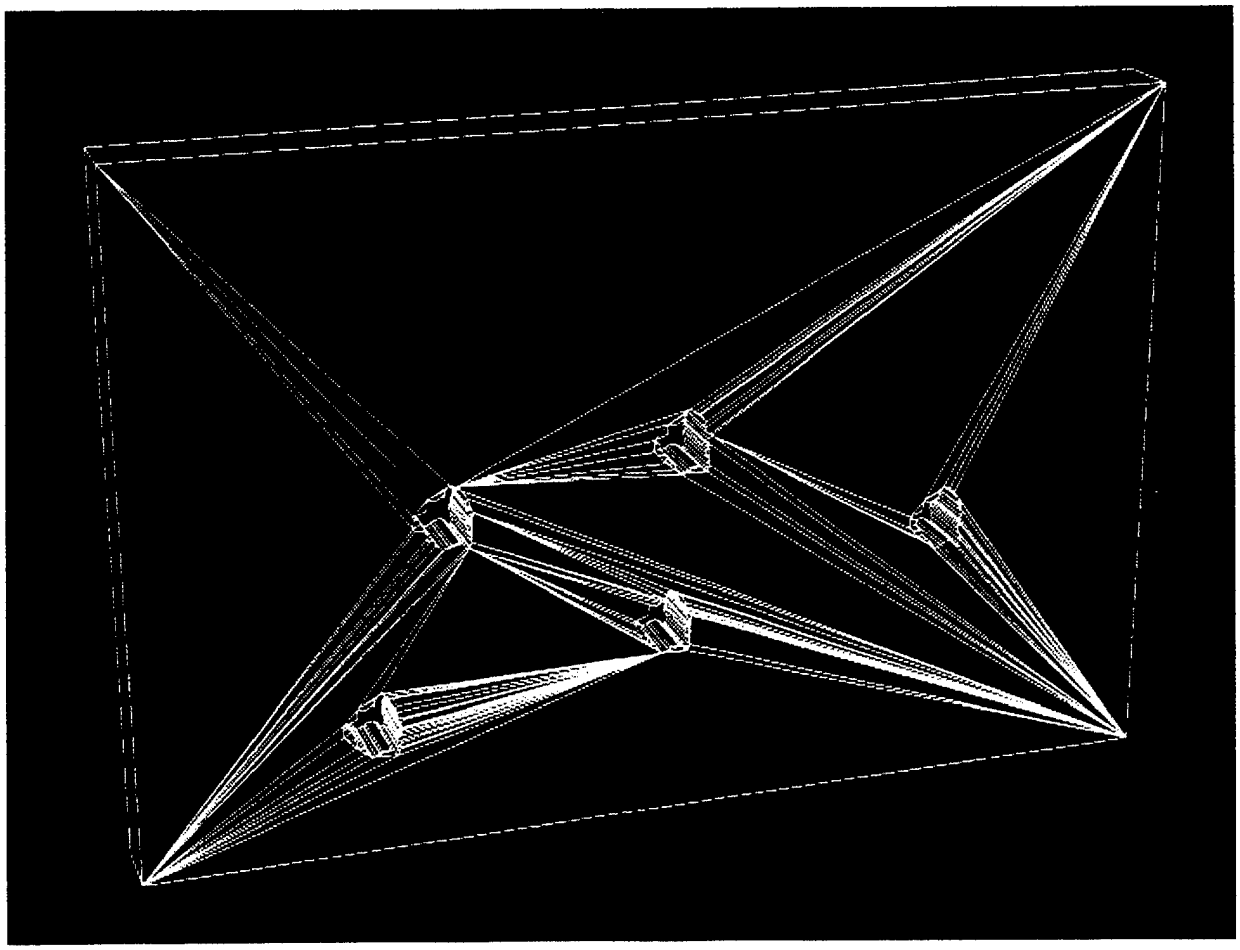
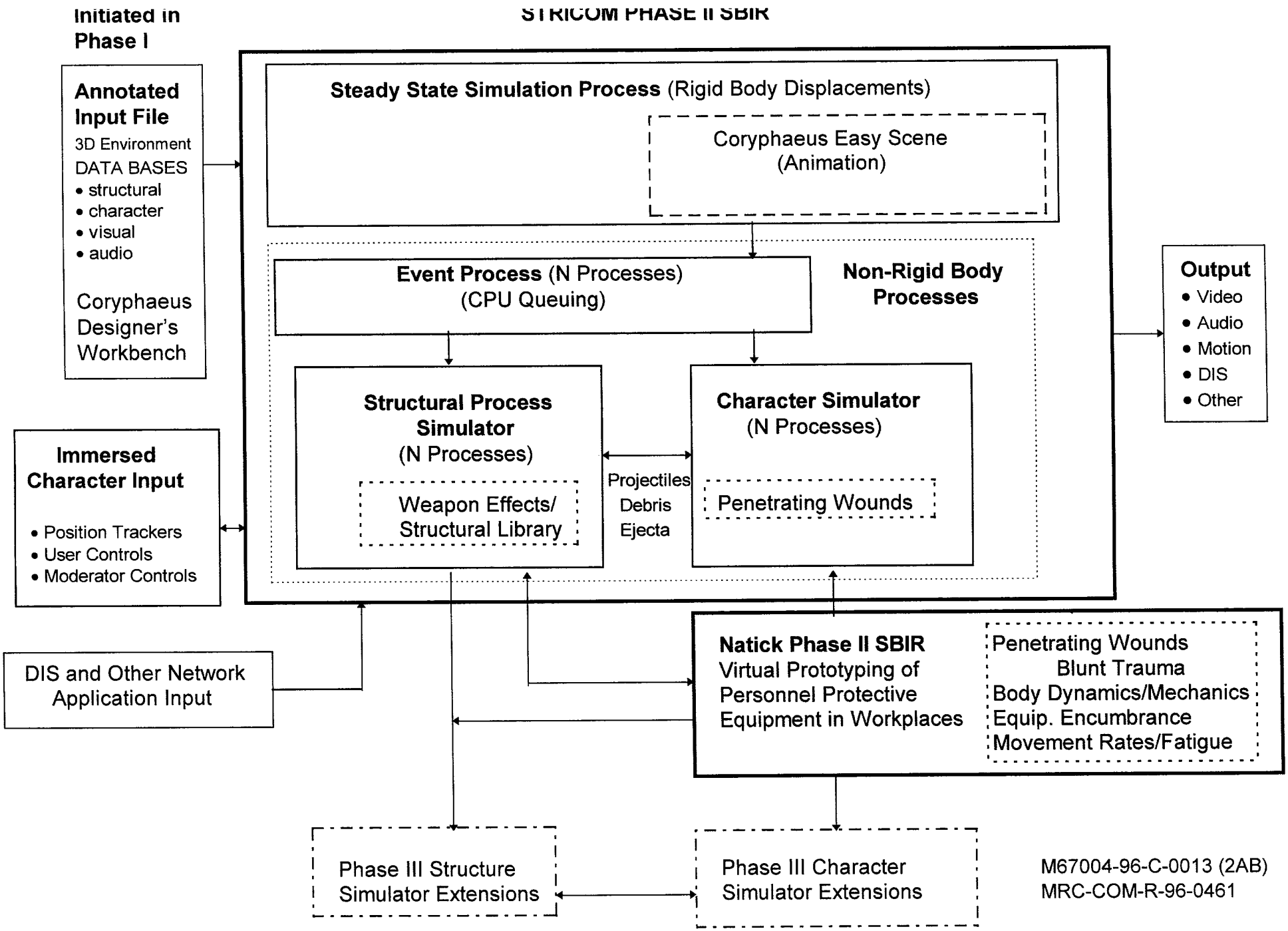


Figure 24. Wall with Five Bullet Holes – 807 Polygons

In the attached figure, the grayed areas indicate the focus of our effort and extensions to the state-of-the art. The non-grayed areas indicate existing off-the-shelf solutions that will be exploited and adapted for our purposes. Three dimensional visual representations of buildings in general and the McKenna MOUT site in particular are


Initiated in Phase I
STRICOM PHASE II SBIR
Annotated Input File
3D Environment DATA BASES
• structural
• character
• visual
• audio
Coryphaeus Designer's Workbench
Steady State Simulation Process (Rigid Body Displacements)
Coryphaeus Easy Scene (Animation)
Event Process (N Processes) (CPU Queuing)
Non-Rigid Body Processes
Structural Process Simulator (N Processes)
Weapon Effects/ Structural Library
Projectiles
Debris
Ejecta
Character Simulator (N Processes)
Penetrating Wounds
Output
• Video
• Audio
• Motion
• DIS
• Other
Immersed Character Input
• Position Trackers
• User Controls
• Moderator Controls
DIS and Other Network Application Input
Natick Phase II SBIR
Virtual Prototyping of Personnel Protective Equipment in Workplaces
Penetrating Wounds
Blunt Trauma
Body Dynamics/Mechanics
Equip. Encumbrance
Movement Rates/Fatigue
Phase III Structure Simulator Extensions
Phase III Character Simulator Extensions
M67004-96-C-0013 (2AB)
MRC-COM-R-96-0461

plentiful. However, models suitable for real-time fly-through visual simulations are not as readily available. A baseline model of the McKenna Town Hall building was developed in Phase I. A simulation using this model can accommodate simple animation involving simple translation and rotation of objects. The entire McKenna MOUT site can be developed in a similar manner if necessary.

The problem for the effects we wish to simulate are threefold in terms of extensions to the state of the art. First, we need to allow "atomic" elements of the simulation to deform in a non-rigid manner (e.g., fragment, crush, bend, etc.). Existing visual simulations can not do this since they merely render visible surfaces. Second we need to employ data bases with fundamentally different spatial and temporal resolutions to accomplish the above. Further, many of these effects occur in parallel and must be appropriately queued to the number and speed of the hardware/processors being used.

Finally, various *decisions* must be made. First, what do we need to show in order to make the simulation visually compelling. In addition to visuals, what level of audio or other non-visual cues are necessary. Second, what don't we have to show but never-the-less need to know. The "atomic" or fundamental elements of the simulation must be also be identified. This last issue involves various trade-offs in terms of simulating rigid-body and non-rigid body processes, the number of polygons in the visual simulation, and real-time performance. Also what is the appropriate level of approximation and packaging of the physical processes being simulated.

After wrestling with these problems, it became apparent that the weapons effects models cannot be separated from the visualization strategy. The visualization of these effects is the real-time constraint. The physical effects models would therefore have to be tailored both in terms of level of approximation and in terms of model parameters so that that a minimum number of atomic elements and polygons occur in the visualization so that the visualization of weapon effects can be done in real-time.

For example, in determining the effect of a shot gun blast through an interior wall. The simulation will not perform the detailed calculations that describe the interaction of the projectiles with each of the wall materials. Rather, a library of interior wall constructions will be provided with parametric solutions specific to the projectile and wall construction combination. The parametric solutions will be compared with detailed calculations and experiments in Phase II. For some scenarios we have already done this in Phase I.

Similarly, debris fields will require physical models that describe particulate density, and visual transmittence as a function of time. For debris fields with casualty producing debris (e.g., glass shards), it is not practical (due to limitations associated with the 3D visualization methodology of debris fields) to track each shard but rather develop a stochastic model which describes the distribution of ejecta and ejecta velocities, and performs a random draw to determine striking conditions on personnel. Once the

striking conditions are prescribed, wound calculations (to the level of fidelity required for non-medical purposes) can be done in real-time.

We already have a mature methodology to perform wound analysis on personnel that operates faster than real-time and we merely have to adapt this. It is likely that most of the software we will need will be developed by us either in a new contract from Natick/Simulation Technologies or Natick's Phase II SBIR. Since we already know what to do here relative to penetrating wounds we did not waste Phase I effort belaboring this issue but we will describe our approach in the Phase II proposal. As part of the STRICOM effort we only expect to focus on penetrating wounds. In the Natick Phase II effort we will be focusing on other types of trauma which can be exported to the STRICOM simulation. For medical simulation we can provide interfaces with the DARPA surgical simulators that are already under development and which eventually will be operating in real-time (there is no point in reinventing the wheel). In other words when someone approaches a casualty and looks down within a prescribed range (you would switch to the DARPA simulators).

3.6 STRICOM Phase II SBIR Software Architecture

Input files will be monolithic (i.e. containing both visual and structural information) to maintain synchronization of the data as the number and size of the data bases grow. We expect that the input format would permit interface with other visualization tools.

The overall structure will be designed to execute as a single UNIX process. However, in an environment with more than one processor available, various secondary processes can be forked to permit better real time performance. We will build a hardware environment with multiple SGI Indigos and a SGI High End Impact for development purposes. Toward the end of the effort we will export the software to a STRICOM or Natick hardware environment with higher end platforms to verify real-time performance and visual fidelity (this will require someone from our organization to be resident at Natick and/or STRICOM for up to two months at the end of the effort.

We intend to provide an environment with multiple characters interacting (initially two, this is merely limited by the number of processors and input devices available). Two characters will allow us to confront most of the fundamental problems without spending a lot of funding on hardware.

For any simulation system there has to be a core structure which maintains the real time state of the simulated environment. Using *EasyScene*™ from *Coryphaeus*™ *Software* will provide a foundation on which to build the "Steady State Simulation Process". *EasyScene* will provide real time video output from an SGI platform. *The Steady State Simulation Process* controls all of the routine simulation effects which only require translation and rotation of objects (Rigid Body Motion) contained in the data base.

An "event" is defined to be any occurrence in the simulation which *may* physically alter one or more "atomic structures" in the data base (Non-Rigid Body Deformation). An atomic structure is defined to be one object or group in the data base which has a defined structural composition (e.g. a window or a wall panel).

When the Steady State Process detects that an event has occurred, the event is queued for handling by an "Event Process." There may be more than one Event Process available to handle events, depending on the system resources available. Some Event Processes will require a dedicated CPU to maintain real time performance.

The function of the Event Process is to evaluate the nature of the event and the data base environment in which it occurred. From the center of the event, the environment is evaluated and any atomic objects which may be affected by the event are queued. The interaction of the object and the event not only affects the object, but may alter subsequent objects that are being checked. This permits shadowing effects for blasts and the determination of penetration effects of firearms.

The Event Process determines those objects which will be affected first (the objects closest to a blast or adjacent objects in the line of fire) and queue the appropriate combinations of "Structural Processes" and "Character Processes." After these processes have evaluated the effect on themselves and the affect of the event, the data is returned to the Event Process. If the event can still affect another object, the next object is queued for evaluation. This continues until the effect of the event dissipates.

The Structural Process has two functions, determine the effect of an event, and create a new visual representation of the object as it would appear after the event. The visual representation is made up of two parts, the permanent representation and any transient effects (e.g. dust, smoke, debris). The updated visual representation is passed back to the Steady State Process for rendering.

The base requirements for the Character Process are the same as the Structural Process with the addition of character animation and biomechanical functions. These biomechanical functions would be emphasized in the Natick Phase II SBIR and developed to a much higher level of sophistication and fidelity than in the STRICOM effort. Similarly, the STRICOM effort would emphasize the structural processes to a much higher level of fidelity and sophistication than in the Natick effort. Both the STRICOM and Natick efforts would have to have some minimal level of performance in both the structural and character processes in case both efforts are not funded. That is, both efforts need to be stand-alone products.

4. CONCLUDING REMARKS

The general problem addressed by the first phase of this effort was developing real-time models to describe weapon effects for DIS virtual environments. More specifically, these effects needed to describe building structural integrity and personnel trauma and be suitable for implementation in a DIS virtual environment. The original Phase I proposal was oriented toward large weapons, i.e., weapons that could completely destroy buildings and was decoupled from the virtual representation of these effects. That is, originally, the Phase I effort was conceived of as theoretically developing the appropriate weapon effects algorithms which would then be implemented in Phase II and integrated with virtual representations outside the scope of this effort.

During the course of this effort, after several interchanges with USAIS, it became clear that the most relevant effects, at least initially, were small arms. This change of orientation, in a sense, was fortunate in that many of the fundamental problems associated with representing weapon effects converge in modeling small arm interaction with building materials and personnel. Further, we decided to select a subset of effects (bullet holes, debris and ejecta, and residual penetrators) that are common to all weapon effects and develop models for these effects to a level of maturity beyond what was originally envisioned in the Phase I effort. This was done in order to crystallize the really technically stressing problems associated with the goal of this effort and to highlight what really needed to be done in Phase II.

The basic problem of creating a hole and/or many holes in walls, along with debris and ejecta, and describing in parallel the effect of residual penetrators exiting the wall and striking other walls and/or people, distills many of the technical obstacles that need to be confronted in simulating these and other effects using physically based models in a virtual environment (as opposed to scripts). This set of problems is fundamental and in a sense, the ability to successfully address these problems represents a "litmus test" indicating whether a general solution/approach is feasible.

It became apparent for this application that development of the weapon effects models *per se* was not technically stressing. Well established methodologies and data bases exist to evaluate blast, penetration, and damage to buildings and personnel to varying levels of approximation. At MRC we have a plethora of these models and data bases some of which we have been funded to develop. The methodologies selected for Phase II implementation, as well as their rationale, are discussed in the Phase II proposal for this effort. They represent well established and validated approaches. In particular, the methodology for describing weapon effects on personnel was developed by MRC and software already exists to address this problem.

By the end of Phase I it became apparent that the really technically stressing issues concern *coupling/embedding physically based models in the virtual environment* not

developing real-time algorithms which describe these effects. This orientation is reflected in our Phase II proposal.

We did not devote significant resources in Phase I in areas where we already "had the answer." It is recognized however, because of the visualization "choke" points that implementing models of physical processes in the visual environment would entail, trade-offs needed to be made in terms of the detail with which physical processes could be modeled and the degrees-of-freedom present in interaction with the virtual environment. The work done in Phase I helps to identify where these trade-offs can be made without affecting the utility of the virtual environment as a training and mission rehearsal tool.

5. Listing of MRSAW Source Code

```
$DEBUG
C --- set up interface to dos
      INTERFACE TO INTEGER*2 FUNCTION SYSTEM [C] (STRING)
      CHARACTER STRING[REFERENCE]
      END
C ======================== END INTERFACE SETUP ==============================

C     PROGRAM AIRBULL7.FOR

C     CREATED AUGUST 1996 TO DECSCRIBE THE MOTION OF VAROIUS BULLETS IN
C     AIR. BESIDES THE ACCELERATION DUE TO GRAVITY, THE DECELERATION
C     PRODUCED BY WIND DRAG IS MODELLED BY A FUNCTIONAL FORM

C     A*V**(2-N)

C     WHERE A IS THE AIR DRAG COEFFICIENT AND N IS THE AIR DRAG INDEX.
C     THESE VARIABLES ARE CONSTANT WITHIN REGIONS OF LOCAL BULLET VELOCITY
C     WHICH IS ASSUMED TO STAY IN THE DOMAIN DURING THE FLIGHT PATH. THIS
C     IS VALID FOR TARGET RANGES DEPENDING ON MUZZLE VELOCITY.

C     INPUT DESCRIPTION:

C     V0 = INITIAL BULLET SPEED (MUZZLE VELOCITY) IN FT/SEC
C     DLTAD = BULLET ORIENTATION WITH THE HORIZONTAL IN DEGREES
C     R = TARGET RANGE IN FT. = NORMAL DISTANCE FOR OBLIQUE TARGET
C     A = AIR DRAG COEFFICIENT
C     N = AIR DRAG INDEX
C     VWX = CROSSWIND SPEED IN THE HORIZONTAL DIRECTION
C     VWY = CROSSWIND SPEED IN THE VERTICAL DIRECTION
C     ALPA,BETA,GAMA = DIRECTION COSINES OF THE NORMAL TO THE
C     TARGET WITH RESPECT TO THE BULLET XYZ AXES SYSTEM
C     ER = COEFFICIENT OF RESTITUTION FOR BULLET-TARGET IMPACT
C     RIGG = MOMEMNT OF INERTIA OF THE BULLET ABOUT ITS CM
C     XG = DISTANCE OF THE CENTER OF MASS FROM THE BULLET TIP
C     R1 = DISTANCE OF FRONT TIP FROM THE CENTER OF MASS G
C     R2 = DISTANCE OF REAR TIP FROM THE CENTER OF MASS G

C     ENERGY LOST DUE TO PENETRATION : ARRAY ENRLST(4,4)
C     GBL/MRC EXPERIMENTAL DATA BASE

C     ENRLST(I,J), I = PROJECTILE TYPE ; J = TARGET TYPE
C                    = 1 FOR SLUG          = 1 FOR INTERIOR WALL
C                    = 2 FOR #00LEAD       = 2 FOR INTERIOR DOOR
C                    = 3 FOR #04CU-PLA     = 3 FOR FLOOR
C                    = 4 FOR 9MM           = 4 CEILING

C     NUMBER OF PELLETS (NP) VS RANGE DISTRIBUTION (R)

C     NP=NPELL(NPRJ)*EXP(-ALFA*(R/RREF)*(BEETA-r/R))



      IMPLICIT REAL*8(A-H,O-Z)
      REAL*4 RAN0
      REAL*8 N,NN1,NN2,LB,LF
      REAL*8 L1,M1,N1,L2,M2,N2,L3,M3,N3,LP,MP,NP
      REAL*8 LXP,MXP,NXP,LZP,MZP,NZP
      REAL*8 NC(7)
      INTEGER*2 SYSTEM
      EXTERNAL Y1DOT2,Y2DOT2
```

```
      CHARACTER*15 TARGET(4),PROJ(4)

      DIMENSION ENRLST(4,4),NPELL(4),ALFA(4),BEETA(4),RF(4)

      DIMENSION XT(500),YT(500),TP1X(500),TP1Y(500),TP1Z(500),
     C          TP2X(500),TP2Y(500),TP2Z(500),
     C          CMGX(500),CMGY(500),CMGZ(500)

      DIMENSION VC(8),AC(7)
      COMMON/INITDAT/Y10,Y1D0,Y20,Y2D0
      COMMON/FUNDAT/A,N,VWX,VWY,PI
      COMMON/COMDAT/R
      COMMON/TRANSDAT/VIM,DLTAIM,XQ,YQ,ALPA,BETA,GAMA

      COMMON/SYSINFO/L1,M1,N1,L2,M2,N2,L3,M3,N3

      DATA VC/3600.0,2600.0,1800.0,1370.0,1230.0,970.0,790.0,0.0/

      DATA NC/0.45,0.30,0.0,-1.0,-3.0,-1.0,0.0/

      DATA AC/246.0,801.3,7599.0,1.045E7,1.578E13,1.685E7,21386/

      DATA ENRLST/4.402E6,5.66722E6,5.24832E6,5.66722E6,
     C            6.41025E6,8.72053E6,11.76933E6,19.73025E6,
     C            6.9828E6,9.14848E6,14.20848E6,24.64473E6,
     C            0.4651E6,0.86822E6,0.5472188E6,0.70922E6/

      DATA TARGET/' INT. WALL',' INT. DOOR',' FLOOR',' CEILING'/

      DATA PROJ/' SLUG',' #00 LEAD','#4 CU-PLA','9MM BULLET'/

      DATA NPELL/1,16,27,1/

      DATA ALFA/0.0D0,116.774D0,131.946D0,0.0D0/
      DATA BEETA/0.0D0,0.026D0,0.025D0,0.0D0/
      DATA RF/0.0D0,120.0D0,120.0D0,0.0/

C     CONSTANTS


      PI=DATAN(1.0D0)*4.0D0
      DTR=PI/180.0D0


C     DEFAULT VALUES

      NDEBUG=0


C      CALL TLOCATE(7,0)
C      CALL CLS(2,0)

      I=SYSTEM('CLS'C)

      WRITE(*,*) ' Executing AIRBULL7 Code .....'
      WRITE(*,*) ' '

C      PAUSE ' Please enter any key to continue....'
```

```
      I=SYSTEM('TYPE BULLSC.INF'C)

      READ(*,*) SC

      I=SYSTEM('CLS'C)

      WRITE(*,*) ' Please enter bullet weight in grams..'
      READ(*,*) WTGM

C     CONVERT WEIGHT IN GRAINS

      WT=WTGM*15.43236D0

      WRITE(*,*) ' Please enter bullet diameter in inches...'
      READ(*,*) DI


C     DETEREMINATION OF BALLISTIC COEFFICIENT BC FOR THE ABOVE BULLET
C     THIS RESULT IS EMPIRICAL: FROM MODERN PRACTICAL BALLISTICS: PEJSA

      RNUM=(DI+0.50D0)/DI**2
      RDIN=415.0D0*(8-2.0D0*SC+SC**2)
      BC=WT*RNUM/RDIN

C      CALL TLOCATE(7,0)
C      CALL CLS(2,0)


      I=SYSTEM('CLS'C)


      WRITE(*,302) SC
302   FORMAT(' Bullet Shape Class = ',f6.2,/)

      WRITE(*,304) WTGM
304   FORMAT(' Bullet Weight ',G10.4,' GM',/)

      WRITE(*,306) DI
306   FORMAT(' Bullet Diameter ',G10.4,' IN',/)

      WRITE(*,308) BC
308   FORMAT(' Computed Ballistic Coefficient = ',G10.4,/)

      OPEN(UNIT=4,FILE='AIRBULL7.IN')

      READ(4,*) V0,DLTAD,R


C     DETERMINE MEYEVSKI PARAMETERS A AND N

      DO 322 I=1,7
      TST=(VC(I+1)-V0)/(VC(I)-V0)
      NCNT=I
      IF(TST.LE.0.0D0) GOTO 324
322   CONTINUE

      WRITE(*,*) ' MUZZLE VELOCITY IS OUTSIDE DATA RANGE..'
      WRITE(*,*) ' PROGRAM TERMINATED ...'
      STOP
```

```
324   CONTINUE

      A=1.0D0/AC(NCNT)
      N=NC(NCNT)

C     MODIFY A TO INCLUDE BALLISTIC COEFFICIENT BC IN INGALLS'S TABLE

      A=A/BC

C      READ(4,*) A,N

      WRITE(*,326) A
326   FORMAT(' Computed Meyevski Parameter A = ',G10.4,/)

      WRITE(*,328) N
328   FORMAT(' Computed Meyevski Parameter N = ',G10.4,/)

      PAUSE ' Please enter any key to continue....'

C      CALL TLOCATE(7,0)
C      CALL CLS(2,0)


      I=SYSTEM('CLS'C)

      READ(4,*) VWX,VWY
C      READ(4,*) ALPA,BETA,GAMA
      READ(4,*) THZD,THYD

C     COMPUTE ALPA, BETA AND GAMA

      THZR=THZD*DTR
      THYR=THYD*DTR

      ALPA=-DCOS(THZR)*DCOS(THYR)
      BETA=-DSIN(THZR)
      GAMA=-DCOS(THZR)*DSIN(THYR)

      READ(4,*) ER
      READ(4,*) RHO,BM,BR,LB,LF
      READ(4,*) R1,R2
      READ(4,*) RIGG,XG


      DLTAR=DLTAD*DTR

      Y10=0.0D0
      Y20=0.0D0
      Y1D0=V0*DCOS(DLTAR)
      Y2D0=V0*DSIN(DLTAR)


C     CROSSWIND CORRECTION OF INITIAL CONDITIONS

C      Y1D0=Y1D0-VWX
C      Y2D0=Y2D0-VWY

C      DLTAR=DATAN(Y2D0/Y1D0)
```

```
      DLTAD=DLTAR/DTR

C     END CROSSWIND CORRECTION


      TANDLTA=DTAN(DLTAR)

      TMAX=20.0D0*R/V0

      NT=500


      DT=TMAX/(NT-1)

      OPEN(UNIT=8,FILE='TY1EX.PLT')
      OPEN(UNIT=9,FILE='TY2EX.PLT')
      OPEN(UNIT=10,FILE='Y1Y1DEX.PLT')
      OPEN(UNIT=11,FILE='Y1Y2DEX.PLT')
      OPEN(UNIT=12,FILE='Y1Y2EX.PLT')

C      WRITE(8,*) NT
C      WRITE(9,*) NT
C      WRITE(10,*) NT
C      WRITE(11,*) NT



      F0=V0**N/A

      NN1=N-1.0D0
      NN2=N-2.0D0

C     G = GRAVITATIONAL ACCELERATION IN FT/S**2

      G=32.2D0


      I=SYSTEM('CLS'C)

C      CALL TLOCATE(7,0)
C      CALL CLS(2,0)

      WRITE(*,*) ' Continue Executing AIRBULL7 Code .....'
      WRITE(*,*) ' '

      WRITE(*,2) R
2     FORMAT(' TARGET RANGE = ',G10.3,' FT ',/)

      WRITE(*,4) V0
4     FORMAT(' INITIAL BULLET SPEED = ',G10.3,' FT/SEC ',/)

      WRITE(*,6) DLTAD
6     FORMAT(' INITIAL BULLET ORIENTATION WITH HORIZONTAL ',F6.3,' DEG
     C. ',/)

      WRITE(*,47)
47    FORMAT(' CROSSWIND SPEED: ')

      WRITE(*,48) VWX
48    FORMAT(' HORIZONTAL DIRECTION = ',G10.3,' FT/SEC ')
```

```
      WRITE(*,52) VWY
 52   FORMAT(' VERTICAL DIRECTION = ',G10.3,' FT/SEC ')

      PAUSE ' ENTER ANY KEY TO CONTINUE .....'



      I=SYSTEM('CLS'C)

C      CALL TLOCATE(7,0)
C      CALL CLS(2,0)


C       RESULTS FROM SHORT DISTANCE LINEAR THEORY

      WRITE(*,8)
 8    FORMAT(' RESULTS FROM SHORT DISTANCE LINEAR THEORY...',/)


      DO 10 I=1,NT

      T=(I-1)*DT

      Y1=(F0/N)*(1.0D0-(1.0D0-NN1*V0*T/F0)**(N/NN1))

      Y2=-(G*F0/(V0**2*NN2))*(Y1+(F0/(2*NN1))*
     C     ((1.0D0-N*Y1/F0)**(2.0D0*NN1/N)-1.0D0))

C     Y2 CORRECTION FOR SHOT ANGLE DLTA: ADDED: AKC

      Y2=Y2+Y1*TANDLTA

      Y1D=V0*(1.0D0-N*Y1/F0)**(1.0D0/N)
      Y2D=(Y1D*G*F0/(V0**2*NN2))*((1.0D0-N*Y1/F0)**(NN2/N)-1.0D0)

      VEL=DSQRT(Y1D**2+Y2D**2)

C     CROSSWIND CORRECTION FOR LOCATION

      Y1COR=VWX*T
      Y2COR=VWY*T-Y1*VWX/V0

      Y1=Y1+Y1COR
      Y2=Y2+Y2COR

      TH=ATAN(Y2D/Y1D)
      THD=TH/DTR

C     TEST TARGET HIT BY THE CONDITION THAT

C          ALPA*Y1+BETA*Y2+R=0

      HIT=ALPA*Y1+BETA*Y2+R

C      IF(Y1.GE.R) GOTO 14

      IF(HIT.LE.0.0D0) GOTO 14
```

```
      WRITE(8,*) T,Y1
      WRITE(9,*) T,Y2
      WRITE(10,*) Y1,Y1D
      WRITE(11,*) Y1,Y2D
      WRITE(12,*) Y1,Y2

      TSV=T
      Y1SV=Y1
      Y2SV=Y2
      VELSV=VEL
      THDSV=THD

 10   CONTINUE


      WRITE(*,12)
 12   FORMAT(' TARGET NOT YET REACHED BY THE BULLET....',/)

      GOTO 18

 14   CONTINUE


      WRITE(*,32)

 32   FORMAT(' TARGET HIT LOCATION: ',/)

      WRITE(*,34) Y1SV
 34   FORMAT(' HORIZONTAL DISTANCE = ',G10.4,' FT',/)

      WRITE(*,36) Y2SV
 36   FORMAT(' VERTICAL DISTANCE = ',G10.4,' FT',/)


      WRITE(*,38) TSV
 38   FORMAT(' TIME TO HIT THE TARGET = ',G10.4,' SEC',/)


      WRITE(*,42) VELSV
 42   FORMAT(' IMPACT VELOCITY AT TARGET = ',G10.4,' FT/SEC',/)


      WRITE(*,44) THDSV
 44   FORMAT(' BULLET ORIENTATION WITH HORIZONTAL = ',G10.4,' DEG',/)


      PAUSE ' ENTER ANY KEY TO CONTINUE .....'


      I=SYSTEM('CLS'C)

C      CALL TLOCATE(7,0)
C      CALL CLS(2,0)


 18   WRITE(*,22)
 22   FORMAT(' RESULTS FROM EXACT NONLINEAR ANALYSIS ',/)
```

```
      CALL RKNLR2(Y1DOT2,Y2DOT2,DT,0.0D0,TMAX,IRR)


      PAUSE ' Enter any key to continue. ....'

      I=SYSTEM('CLS'C)

      I=SYSTEM('TYPE PROJ.INF'C)

      READ(*,*) NPRJ

C     GENERATING 3-D PLOT FILES FOR THE BULLET LOCATIONS

      OPEN(UNIT=7,FILE='Y1Y2.PLT')
      OPEN(UNIT=5,FILE='Y1Y1D.PLT')
      OPEN(UNIT=6,FILE='Y1Y2D.PLT')

      NDAT=0

      DO 100 I=1,500
      READ(7,*,END=102) XT(I),YT(I)
      READ(5,*) DUM,RY1D
      READ(6,*) DUM,RY2D

      NDAT=NDAT+1

      RVEL=DSQRT(RY1D**2+RY2D**2)

      CMGX(I)=XT(I)
      CMGY(I)=YT(I)
      CMGZ(I)=0.0D0

C     FOR BULLETS ONLY

      IF(NPRJ.NE.4) GOTO 100

      TP1X(I)=XT(I)+R1*RY1D/RVEL
      TP1Y(I)=YT(I)+R1*RY2D/RVEL
      TP1Z(I)=0.0D0


      TP2X(I)=XT(I)-R2*RY1D/RVEL
      TP2Y(I)=YT(I)-R2*RY2D/RVEL
      TP2Z(I)=0.0D0

100   CONTINUE

102   CONTINUE

      WRITE(*,202) NDAT
202   FORMAT(' DATA POINTS FOR PRE-IMPACT MODE',I4,/)

C     SAVE PLOT FILE AND TERMINATE IF NO HIT OCCURS...

      IF(VIM.EQ.0.0D0) GOTO 674

C     USER PROMT FOR PENETRATION OR RICOCHET MODE

C     CURRENTLY DATA IS NOT AVAILABLE FOR DETERMINING THIS TRANSITION
C     AUTOMATICALLY BY THE CODE. MORE TESTS ARE NEEDED TO ESTABLISH
```

```
C     THESE THRESHOLD SPEEDS FOR VARIOUS TARGETS AND PROJECTILES

      I=SYSTEM('CLS'C)

      I=SYSTEM('TYPE TARGET.INF'C)

      READ(*,*) NTGT

      I=SYSTEM('CLS'C)

      WRITE(*,*) ' YOU HAVE ENTERED THE PROJECTILE-TARGET PAIR:'

      WRITE(*,404) PROJ(NPRJ)
404   FORMAT( ' PROJECTILE: ',15A,/)

      WRITE(*,406) TARGET(NTGT)
406   FORMAT( ' TARGET: ',15A,/)


      VIN=VELSV

      WRITE(*,412) VIN
412   FORMAT(' IMPACT VELOCITY = ',G12.3,' FT/SEC',/)


C     THRESHOLD VELOCITY VPEN FOR PENETRATION

C     ENTERING TARGET PENETRATION MODE

C     CALCULATE VPEN ENERGY LOSS FROM DATABASE

      ELOSS=ENRLST(NPRJ,NTGT)

C     WEIGHT IS USED INSTEAD OF MASS AS ENERY LOSS DATA ARRAY
C     ENRLST HAS BEEN MODIFIED ACCORDINGLY


      EIN=0.5D0*WTGM*VIN**2

      VPEN=DSQRT(2.0D0*ELOSS/WTGM)

      WRITE(*,662) VPEN
662   FORMAT(' THRESHOLD VELOCITY FOR PENETRATION: ',G10.3,' FT/SEC',/)

C      WRITE(*,*) ' PLEASE ENTER THE THRESHOLD VELOCITY FOR PENETRATION'

C      READ(*,*) VPEN

      I=SYSTEM('CLS'C)

C     DETERMINE WHICH MODE WILL OCCUR: RICOCHET MODE OR PENETRATION MODE


      IF(VIN.LT.VPEN) GOTO 408

      WRITE(*,*) ' TARGET PENETRATION OCCURRED...'


      EIN=0.5D0*WTGM*VIN**2
```

```
      VEX=VIN*DSQRT(1.0D0-ELOSS/EIN)

      I=SYSTEM('CLS'C)

      WRITE(*,422) VIN
422   FORMAT(/,' TARGET ENTRY VELOCITY = 'G12.3,'FT/S',/)

      WRITE(*,423) VEX
423   FORMAT(' TARGET EXIT VELOCITY = 'G12.3,'FT/S',/)

C     ALSO ENTER

C     TARGET DAMAGE AREA: POINT FOR SLUGS AND BULLETS (NPRJ=1,4)
C                         AREA FOR BUCKSHOT (NTGT=2,3)

      NTST=(NPRJ-1)*(NPRJ-4)

      IF(NTST.NE.0) GOTO 502

C     ONE SHOT DAMAGE FOR SLUG AND BULLETS

      WRITE(*, 504) Y2SV
504   FORMAT(' TARGET HIT AREA = ',G10.2,' ABOVE HORIZONTAL',/)

      GOTO 506

502   CONTINUE

C     FOR BUCKSHOT MODE ONLY

C     RUN RANDOM NUMBER GENERATOR FOR PAIRS OF NUMBER OF PELLETS

      NPEL=NPELL(NPRJ)

      WRITE(*,516) NPEL
516   FORMAT(' NUMBER OF PELLETS HITTING THE TARGET: ',I3,/)

C     SPRANG = SPRAY ANGLE IN RADIAN

      SPRANG=DATAN(BEETA(NPRJ))

      SPRANGD=SPRANG/DTR

      WRITE(*,512) SPRANGD
512   FORMAT(' PELLET SPRAY ANGLE = ',G10.3,'(DEG)',/)


C     SPRAD=SPRAY RADIUS ON THE THE TARGET

      SPRAD=R*BEETA(NPRJ)

      WRITE(*,611) SPRAD

611   FORMAT(' SPRAY RADIUS ON THE TARGET = ',G10.3,/)

      PAUSE ' ENTER TO CONTINUE....'

      I=SYSTEM('CLS'C)
```

```
      WRITE(*,518)
 518  FORMAT(' HIT LOCATIONS:',/)

      WRITE(*,522)
 522  FORMAT(T3,' RADIAL DISTANCE',T25,'ANGULAR LOCATION',/)

C     READ ISEED FOR FROM FILE FOR RANDOM NUMBER GENERATION

      OPEN(UNIT=11,FILE='RANDOM.SED')

      READ(11,*) ISEED

      CLOSE(UNIT=11)

      OPEN(UNIT=11,FILE='RANDOM.SED')

C     PELLET DISTRIBUTION VS RADIAL DISTANCE
C     STORED IN PELLET.DIS

      OPEN(UNIT=14,FILE='PELLET.DIS')

      SMR1=0.0D0

      DO 510 I=1,NPEL

      RNPEL=NPEL*1.0D0

      RI=I*1.0D0

C     FOLLOWING EQUATION RESULTS FROM THE INVERSION OF
C     PELLET DISTRIBUTION FUNCTION AS A FUNCTION OF R,r

      SMR2=R*(BEETA(NPRJ)-(RF(NPRJ)/R)*DLOG(RNPEL/RI)/ALFA(NPRJ))

      WRITE(14,*) I,SMR2

      RAN=RAN0(ISEED)
      RAD=SMR1+(SMR2-SMR1)*RAN
      ANG=360.0D0*RAN

      WRITE(*,524) RAD,ANG

      SMR1=SMR2

 510  CONTINUE

 524  FORMAT(T5,G12.4,T22,G12.4)

C     REWRITE ISEED FOR NEXT USE

      WRITE(11,*) ISEED


C     TARGET TYPE
C     PROJECTILE TYPE
C     TARGET RANGE
C     MUZZLE VELOCITY

C     FINISH TARGET PENETRATION MODE: WRITE PLOT DATA AND EXIT
```

```
      GOTO 532


 408  CONTINUE

      WRITE(*,*) ' TARGET PENETRATION DID NOT OCCUR...'

      WRITE(*,204)
 204  FORMAT(/,' ENTERING PROJECTILE REBOUND MODE....',/)


      PAUSE ' ENTER ANY KEY TO CONTINUE......'


      I=SYSTEM('CLS'C)

C      CALL TLOCATE(7,0)
C      CALL CLS(2,0)


C     ORIGIN WILL NOW BE SHIFTED TO Q TO DETERMINE THE PATH
C     DUE TO BULLET RICOCHETTING


C     COORDINATES CONVERTED TO USER FRAME

C     DIRECTION COSINES OF NEW AXES SYSTEM WITH ORIGIN AT Q

C     X2-AXIS: (L1,M1,N1)
C     Y2-AXIS: (L2,M2,N2)
C     Z2-AXIS: (L3,M3,N3)


C     CALCULATION OF IMPACT KINETICS

C     (ALPA,BETA,GAMA) =  D.C. OF THE NORMAL TO THE TARGET WRT.
C     BULLET X-Y FRAME

      DLTAIMR=DLTAIM*DTR

      L1=DCOS(DLTAIMR)
      M1=DSIN(DLTAIMR)
      N1=0.0D0

      WRITE(*,*) ' DIRECTION COSINES OF THE IMPACT DIRECTION:'
      WRITE(*,352) L1,M1,N1
 352  FORMAT(3G12.5,/)

      WRITE(*,*) ' DIRECTION COSINES OF THE IMPULSE DIRECTION:'
      WRITE(*,352) ALPA,BETA,GAMA

C     INCIDENT ANGLE WITH THE IMPULSE DIRECTION

      THI=DACOS(-L1*ALPA-M1*BETA-N1*GAMA)
      THID=THI/DTR

      WRITE(*,354) VIM
 354  FORMAT(' INCIDENT VELOCITY: ',G12.5,'FT/SEC',/)
```

```
      WRITE(*,356) THID
  356 FORMAT(' INCIDENT ANGLE ',G10.5,
     C       ' DEG. WITH THE IMPULSE DIRECTION',/)

      WRITE(*,358) ER
  358 FORMAT(' Coeff. of Restitution ',G12.5,/)

      TFAC=DTAN(THI)

      THO=DATAN(TFAC/ER)
      THOD=THO/DTR

      WRITE(*,362) THOD
  362 FORMAT(' REBOUND ANGLE ',G10.5,
     C       ' DEG. WITH THE IMPULSE DIRECTION',/)

      VP=VIM*DSIN(THI)/DSIN(THO)

      WRITE(*,364) VP
  364 FORMAT(' REBOUND VELOCITY: ',G12.5,' FT/SEC')


      RIMP=BM*VIM*DSIN(THI+THO)/DSIN(THO)


      WRITE(*,366) RIMP
  366 FORMAT(' Impulse = ',G12.5,'(GM.FT/SEC)',/)

      VRAT=VIM/VP
      FACIMP=RIMP/(BM*VP)
      LP=VRAT*L1+FACIMP*ALPA
      MP=VRAT*M1+FACIMP*BETA
      NP=VRAT*N1+FACIMP*GAMA

      CS=L1*ALPA+M1*BETA+N1*GAMA
      SN=DSQRT(1.0D0-CS**2)

      L3=(M1*GAMA-N1*BETA)/SN
      M3=(N1*ALPA-L1*GAMA)/SN
      N3=(L1*BETA-M1*ALPA)/SN

      L2=M3*NP-N3*MP
      M2=N3*LP-L3*NP
      N2=L3*MP-M3*LP

C     DIRECTION COSINES OF THE NEW AXES SYSTEM AT Q
      WRITE(*,*) ' DIRECTION COSINES OF THE NEW AXES SYSTEM AT THE IMPAC
     CT LOCATION'

      WRITE(*,368) LP,MP,NP
  368 FORMAT( /,' LP, MP, NP = ',3G12.5)

      WRITE(*,372) L2,M2,N2
  372 FORMAT( ' L2, M2, N2 = ',3G12.5)

      WRITE(*,374) L3,M3,N3
  374 FORMAT( ' L3, M3, N3 = ',3G12.5)


C     TRANSFER TO BULLET RICOCHET MODE OR SPHERE REBOUND MODE
```

```
C     ACCORDING AS TH VALUE OF NPRJ EQUAL TO 4 OR 1

      IF(NPRJ.EQ.4) GOTO 544

C     SPHERE REBOUND MODE


C     ALSO ENTER SPHERE REBOUND MODE CALCULATION

C     RUN RK2 FOR DIFFERENT MUZZLE VELOCITY AND SHOT ANGLE

C     PERFORM COORDINATE TRANSFORMATION

C     STORE DATA


      THP=DACOS(MP)
      SNP=DSIN(THP)

      LZP=NP/SNP
      MZP=0.0D0
      NZP=-LP/SNP

      LXP=-NZP
      MXP=0.0D0
      NXP=LZP

C     NEW INITIAL CONDITIONS

      Y10=0.0D0
      Y1D0=VP*SNP
      Y20=0.0D0
      Y2D0=VP*MP

C     RESET TARGET DIRECTION COSINES
C     SET TO RETURN TO THE SHOOTING POINT

      ALPA=1.0D0
      BETA=0.0D0
      GAMA=0.0D0

      RM=R

      CLOSE(UNIT=7)

      CALL RKNLR2(Y1DOT2,Y2DOT2,DT,0.0D0,TMAX,IRR)


C     GENERATING 3-D PLOT FILES FOR THE SLUG LOCATIONS

      OPEN(UNIT=7,FILE='Y1Y2.PLT')
      OPEN(UNIT=5,FILE='Y1Y1D.PLT')
      OPEN(UNIT=6,FILE='Y1Y2D.PLT')

      NDATT=0

      DO 605 I=1,500
      READ(7,*,END=607) XT(I),YT(I)
      READ(5,*) DUM,RY1D
```

```
      READ(6,*) DUM,RY2D

      NDATT=NDATT+1

      RVEL=DSQRT(RY1D**2+RY2D**2)
C      CMGX(I)=XT(I)
C      CMGY(I)=YT(I)
C      CMGZ(I)=0.0D0

 605  CONTINUE


 607  WRITE(*,612) NDATT
 612  FORMAT(' DATA POINTS FOR POST-IMPACT MODE',I4,/)


      L1=LXP
      M1=MXP
      N1=NXP

      L2=0.0D0
      M2=0.0D0
      N2=0.0D0

      L3=LZP
      M3=MZP
      N3=NZP

      CALL SYSCHG(XT,YT,CMGX,CMGY,CMGZ,XG0,YG0,ZG0,NDATT,NDAT)

C     BYPASS BULLET RICOCHET MODE
C     TRANSFER TO FINAL FILE WRITING FOR PLOT

      GOTO 532

 544  CONTINUE

C     BULLET RICOCHET MODE

C     RESET RM TO R = RANGE

      RM=R

C     OPEN AND WRITE DATA FILE FOR THE BULLET RICOCHET CODE BULLPEN


      PAUSE ' Enter any key to continue...'


      I=SYSTEM('CLS'C)

C      CALL CLS(7,0)
C      CALL TLOCATE(2,0)


C     ENTERING RICOCHET MODE AFTER REBOUND

C     RIGG = MOMENT OF INERTIA ABOUT CENTER OF MASS G

C     XG = DISTANCE OF C.G. G FROM THE BULLET TIP
```

```
C 284   RIGG=RHO*(LB**3/12.0D0+LF*LB**2/4.0D0)

284    CONTINUE

C      THDN = BULLET ANGULAR VELOCITY DUE TO IMPACT

       THDN=RIMP*XG*DSIN(THI)/RIGG

       OPEN(UNIT=2,FILE='BULLRICO.IN')


       WRITE(2,*) RHO,BR
       WRITE(2,*) LB,LF
       WRITE(2,*) A,N,VWX,VWY
       WRITE(2,*) '0.0',VP
       WRITE(2,*) '0.0,0.0'
       WRITE(2,*) '90.0',THDN
       WRITE(2,*) LP,MP,NP
       WRITE(2,*) L2,M2,N2
       WRITE(2,*) L3,M3,N3
C       WRITE(2,*) XQ,YQ,ZQ
C       WRITE(2,*) CONC1,CONC2
       WRITE(2,*) '0.0',TMAX,DT
       WRITE(2,*) NDEBUG
       WRITE(2,*) RM

82     CONTINUE

       CLOSE(UNIT=2)


       WRITE(*,*) ' Bullet Ricochetting Mode after Impact '

       I = SYSTEM('BULLRICO'C)


       WRITE(*,*) ' Bullet Ricochetting Completed ...'

C      OPEN DATA FILE CREATED BY BULRICO2.FOR CODE

C       I=SYSTEM('COPY BXY1.PLT BXY12.PLT'C)
C       I=SYSTEM('COPY BXGD.PLT BXGD2.PLT'C)
C       I=SYSTEM('COPY BXY2.PLT BXY22.PLT'C)
C       I=SYSTEM('COPY BXYG.PLT BXYG2.PLT'C)
C       I=SYSTEM('COPY BXG.PLT BXG2.PLT'C)


       OPEN(UNIT=9,FILE='BXY1.PLT')
       OPEN(UNIT=10,FILE='BXY2.PLT')
       OPEN(UNIT=11,FILE='BXYG.PLT')

C       READ(9,*) NDUM
C       READ(10,*) NDUM
C       READ(11,*) NDUM

       ZQ=0.0D0

       XG0=XQ-L1*R1
       YG0=YQ-M1*R1
```

```
      ZG0=ZQ-N1*R1


      OPEN(UNIT=20,FILE='AIRBULL7.DBG')

      WRITE(20,*) ' D.C. OF NEW COORDINATE AXES SYSTEM AT C.M.'

      WRITE(20,*) ' L1,M1,N1=',L1,M1,N1
      WRITE(20,*) ' L2,M2,N2=',L2,M2,N2
      WRITE(20,*) ' L3,M3,N3=',L3,M3,N3

      WRITE(20,*) ' C.M. LOCATION AT IMPACT:'

      WRITE(20,*) ' XG = ',XG0
      WRITE(20,*) ' YG = ',YG0
      WRITE(20,*) ' ZG = ',ZG0


C     CHANGE TO USER COORDINATE SYSTEM THROUGH SYSCHG


      PAUSE ' Enter any key to continue...'


      I=SYSTEM('CLS'C)

C      CALL CLS(7,0)
C      CALL TLOCATE(2,0)


C     READ COORDINATES OF FRONT TIP IN NEW COORDINATE SYSTEM

      NDATT=0

      NMAX=500

      DO 15 I=1,NMAX
      READ(9,*,END=16) XT(I),YT(I)
      NDATT=NDATT+1
15    CONTINUE


16      WRITE(*,*) ' Bullet Ricochet Phase Data Points ',NDATT

      OL1=L1
      OM1=M1
      ON1=N1

      L1=LP
      M1=MP
      N1=NP

      CALL SYSCHG(XT,YT,TP1X,TP1Y,TP1Z,XG0,YG0,ZG0,NDATT,NDAT)

C     READ COORDINATES OF REAR TIP IN NEW COORDINATE SYSTEM


      DO 25 I=1,NDATT
      READ(10,*) XT(I),YT(I)
```

```
25    CONTINUE

      CALL SYSCHG(XT,YT,TP2X,TP2Y,TP2Z,XG0,YG0,ZG0,NDATT,NDAT)

C     READ COORDINATES OF C.M. IN NEW COORDINATE SYSTEM

      DO 135 I=1,NDATT
      READ(11,*) XT(I),YT(I)
135   CONTINUE

      CALL SYSCHG(XT,YT,CMGX,CMGY,CMGZ,XG0,YG0,ZG0,NDATT,NDAT)

      L1=OL1
      M1=OM1
      N1=ON1

      NDAT=NDAT+NDATT

506   CONTINUE

C     WRITE FINAL OUTPUT FILE OF THE LOCI OF TIP 1, TIP2 AND C.M.
C     IN THE USER COORDINATE SYSTEM

532   CONTINUE

      OPEN(UNIT=16,FILE='CMG.PLT')

      IF(NPRJ.NE.4) GOTO 622

      OPEN(UNIT=12,FILE='FTIP.PLT')
      OPEN(UNIT=14,FILE='RTIP.PLT')

      WRITE(12,62) NDAT
      WRITE(14,62) NDAT

622   WRITE(16,62) NDAT

62    FORMAT(I4)

      DO 45 I=1,NDAT
      WRITE(16,*) CMGX(I),CMGY(I),CMGZ(I)

      IF(NPRJ.NE.4) GOTO 45

      WRITE(12,*) TP1X(I),TP1Y(I),TP1Z(I)
      WRITE(14,*) TP2X(I),TP2Y(I),TP2Z(I)

45    CONTINUE

674   IF(NPRJ.NE.4) GOTO 624

C     WRITING ALL DATA (TIP 1, TIP 2, G)

      OPEN(UNIT=18,FILE='XYZALL.PLT')
```

```
      OPEN(UNIT=19,FILE='TXYZ12.PLT')

      DO 50 I=1,NDAT
      WRITE(18,51) TP1X(I),TP1Y(I),TP1Z(I),
     C             TP2X(I),TP2Y(I),TP2Z(I),
     C             CMGX(I),CMGY(I),CMGZ(I)
 50   CONTINUE
 51   FORMAT(9G12.5)


 C    WRITING TO 3D GRAPHIC FILE

 C    RANGE

 C    TARGET NORMAL

      WRITE(19,*) R
      WRITE(19,*) ALPA,BETA,GAMA
      WRITE(19,62) NDAT


      DO 55 I=1,NDAT
      T=I*DT
      WRITE(19,51) T,TP1X(I),TP1Y(I),TP1Z(I),
     C             TP2X(I),TP2Y(I),TP2Z(I)
 55   CONTINUE

      GOTO 99


 624  CONTINUE


      OPEN(UNIT=18,FILE='XYZCM.PLT')
      OPEN(UNIT=19,FILE='TXYZCM.PLT')

      DO 630 I=1,NDAT

      WRITE(18,51) CMGX(I),CMGY(I),CMGZ(I)

 630  CONTINUE

 C    WRITING TO 3D GRAPHIC FILE

 C    RANGE

 C    TARGET NORMAL

      WRITE(19,*) R
      WRITE(19,*) ALPA,BETA,GAMA
      WRITE(19,62) NDAT


      DO 635 I=1,NDAT
      T=I*DT
      WRITE(19,51) T,CMGX(I),CMGY(I),CMGZ(I)

 635  CONTINUE
```

```
 99     WRITE(*,24)
 24     FORMAT(' Exit AIRBULL7 Code.....')

        STOP

        END


        SUBROUTINE SYSCHG(XI,YI,XO,YO,ZO,XQ,YQ,ZQ,JN,NS)

        IMPLICIT REAL*8(A-H,L-Z)

        INTEGER*4 NS

        DIMENSION XI(500),YI(500),XO(500),YO(500),ZO(500)


        COMMON/SYSINFO/L1,M1,N1,L2,M2,N2,L3,M3,N3

        ZI=0.0D0

 C      KEEP TOTAL SUM NS+JN NOT GREATER ITS ALLOCATED LENGTH OF 500

        JN1=500-NS

        NT=NS+JN

        IF(NT.GT.500) JN=JN1

        DO 10 I=1,JN

        XO(NS+I)=L1*XI(I)+L2*YI(I)+L3*ZI+XQ
        YO(NS+I)=M1*XI(I)+M2*YI(I)+M3*ZI+YQ
        ZO(NS+I)=N1*XI(I)+N2*YI(I)+N3*ZI+ZQ
 10     CONTINUE

        RETURN
        END




        FUNCTION Y1DOT2(X,Y1,Y1D,Y2,Y2D)
        IMPLICIT REAL*8(A-H,O-Z)

        REAL*8 N

        COMMON/FUNDAT/A,N,VWX,VWY,PI


        V=DSQRT(Y1D**2+Y2D**2)


 C      INDEX N FOR RETARDATION FUNCTION SUCH THAT THE DECELERATION
 C      IN AIR IS GIVEN BY
 C      DECL=A*V**(2-N)

 C      VREL = VELOCITY OF BULLET RELATIVE TO AIR
```

```
      XREL=Y1D-VWX
      YREL=Y2D-VWY

      VREL=DSQRT(XREL**2+YREL**2)

C     NO CROSSWIND FORM
C      Y1DOT2=-A*V**(1.0D0-N)*Y1D

C      GAMA1=DATAN(XREL/YREL)

C      GAMA=PI-GAMA1

      Y1DOT2=-A*VREL**(1.0D0-N)*XREL

      RETURN

      END

      FUNCTION Y2DOT2(X,Y1,Y1D,Y2,Y2D)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 N

      COMMON/FUNDAT/A,N,VWX,VWY,PI

      V=DSQRT(Y1D**2+Y2D**2)


C     INDEX N FOR RETARDATION FUNCTION SUCH THAT THE DECELERATION
C     IN AIR IS GIVEN BY
C     DECL=A*V**(2-N)
C     G = ACCELERATION DUE TO GRAVITY IN FT/S**2

      G=32.2D0


C     VREL = VELOCITY OF BULLET RELATIVE TO AIR

      XREL=Y1D-VWX
      YREL=Y2D-VWY

      VREL=DSQRT(XREL**2+YREL**2)

C     NO CROSSWIND FORM
C     Y2DOT2=-G-A*V**(1.0D0-N)*Y2D

C      GAMA=DATAN(YREL/XREL)

C      GAMA=PI-GAMA

      Y2DOT2=-G-A*VREL**(1.0D0-N)*YREL

      RETURN

      END



      SUBROUTINE RKNLR2(FUN1,FUN2,HI,XI,XFF,IRR)
      IMPLICIT REAL*8(A-H,K,O-Z)
```

```
      COMMON/INITDAT/Y10,Y1D0,Y20,Y2D0

      COMMON/COMDAT/R


      COMMON/TRANSDAT/VIM,DLTAIM,XQ,YQ,ALPA,BETA,GAMA


C     IRR=ERROR CODE

      IRR=0


      DTR=DATAN(1.0D0)*4.0D0/180.0D0

C     TEST INPUT DATA

      N=(XFF-XI)/HI
      HI=(XFF-XI)/N
      IF(HI) 12,22,32

      IRR=1

12    WRITE(6,21)
21    FORMAT(' NEGATIVE INCREMENT IN RK2: PROGRAM HALTED')
      RETURN

      IRR=2
22    WRITE(6,31)
31    FORMAT(' ZERO INCREMENT IN RK2: PROGRAM HALTED')
      RETURN

32    CONTINUE

C     OPEN OUTPUT FILES

      OPEN(UNIT=2,FILE='TY1.PLT')
      OPEN(UNIT=3,FILE='TY2.PLT')
      OPEN(UNIT=5,FILE='Y1Y1D.PLT')
      OPEN(UNIT=6,FILE='Y1Y2D.PLT')
      OPEN(UNIT=7,FILE='Y1Y2.PLT')

      N1=N+1

C      WRITE(2,*) N
C      WRITE(3,*) N
C      WRITE(5,*) N
C      WRITE(6,*) N
C      WRITE(7,*) N

C     INITIAL VALUES: TRANSFERRED THROUGH THE COMMON BLOCK COMDAT


C     INTEGRATION OF TWO COUPLED NONLINEAR EQUATION

C           Y1D2=FUN1(X,Y1,Y1D,Y2,Y2D)
C           Y2D2=FUN2(X,Y1,Y1D,Y2,Y2D)

C     BY FOURTH ORDER RUNGE-KUTTA ALGORITHM
```

```
C     SET INITIAL VALUES

      Y1=Y10
      Y2=Y20

      Y1D=Y1D0
      Y2D=Y2D0


      WRITE(2,*) XI,Y1
      WRITE(3,*) XI,Y2
      WRITE(5,*) Y1,Y1D
      WRITE(6,*) Y2,Y2D
      WRITE(7,*) Y1,Y2

      HI2=HI/2.0D0
      HI8=HI/8.0D0
      X=XI


      DO 40 I=1,N

C     CALCULATION OF K1, K2, K3,K4

      K11=HI*FUN1(X,Y1,Y1D,Y2,Y2D)
      K12=HI*FUN2(X,Y1,Y1D,Y2,Y2D)

C     WRITE(*,*) ' K11,K12:',K11,K12

      X1=X+HI2

      Y11=Y1+HI2*Y1D+HI8*K11
      Y21=Y2+HI2*Y2D+HI8*K12

      Y1D1=Y1D+K11/2.0D0
      Y2D1=Y2D+K12/2.0D0

      K21=HI*FUN1(X1,Y11,Y1D1,Y2,Y2D)
      K22=HI*FUN2(X1,Y1,Y1D,Y21,Y2D1)

      Y1D2=Y1D+K21/2.0D0
      Y2D2=Y2D+K22/2.0D0

      K31=HI*FUN1(X1,Y11,Y1D2,Y2,Y2D)
      K32=HI*FUN2(X1,Y1,Y1D,Y21,Y2D2)

      X2=X+HI

      Y12=Y1+HI*Y1D+HI2*K31
      Y22=Y2+HI*Y2D+HI2*K32

      Y1D2=Y1D+K31
      Y2D2=Y2D+K32

      K41=HI*FUN1(X2,Y12,Y1D2,Y2,Y2D)
      K42=HI*FUN2(X2,Y1,Y1D,Y22,Y2D2)

C     NEXT STEP VARIABLES

      Y1=Y1+HI*(Y1D+(K11+K21+K31)/6.0D0)
```

```
      Y2=Y2+HI*(Y2D+(K12+K22+K32)/6.0D0)

C     NEXT STEP DERIVATIVES OF THE VARIABLES


      Y1D=Y1D+(K11+2.0D0*K21+2.0D0*K31+K41)/6.0D0
      Y2D=Y2D+(K12+2.0D0*K22+2.0D0*K32+K42)/6.0D0

      VEL=DSQRT(Y1D**2+Y2D**2)

      TH=DATAN(Y2D/Y1D)
      THD=TH/DTR

      X=X+HI


C     TEST TARGET HIT BY THE CONDITION THAT

C          ALPA*Y1+BETA*Y2+R=0

      HIT=ALPA*Y1+BETA*Y2+R


C      IF(Y1.GE.R) GOTO 42

      IF(HIT.LE.0.0D0) GOTO 42

C     WRITE TO FILES


      WRITE(2,*) X,Y1
      WRITE(3,*) X,Y2

      WRITE(5,*) Y1,Y1D
      WRITE(6,*) Y1,Y2D
      WRITE(7,*) Y1,Y2

      XSV=X
      Y1SV=Y1
      Y2SV=Y2

      VELSV=VEL

      THDSV=THD

 40   CONTINUE


      WRITE(*,24)
 24   FORMAT(' TARGET NOT YET REACHED BY THE BULLET ...',/)

      VIM=0.0D0
      DLTAIM=0.0D0

      GOTO 46

 42   CONTINUE

C     TRANSFER DATA TO THE MAIN ROUTINE THORUGH TRANSDAT
```

```
      XQ=Y1SV
      YQ=Y2SV
      VIM=VELSV
      DLTAIM=THDSV


      WRITE(*,26)
26    FORMAT(' TARGET HIT LOCATION: ',/)

      WRITE(*,48) Y1SV
48    FORMAT(' HORIZONTAL DISTANCE = ',G10.4,' FT',/)

      WRITE(*,52) Y2SV
52    FORMAT(' VERTICAL DISTANCE = ',G10.4,' FT',/)

      WRITE(*,54) XSV
54    FORMAT(' TIME TO HIT THE TARGET = ',G10.4,' SEC',/)

      WRITE(*,56) VELSV
56    FORMAT(' IMPACT VELOCITY AT THE TARGET = ',G10.4,' FT/SEC',/)


      WRITE(*,58) THDSV
58    FORMAT(' BULLET ORIENTATION WITH HORIZONTAL = ',G10.4,' DEG',/)

46    CLOSE(UNIT=2)
      CLOSE(UNIT=3)
      CLOSE(UNIT=5)
      CLOSE(UNIT=6)
      CLOSE(UNIT=7)


      RETURN

      END


C     THIS SUBROUTINE GENERATES RANDOM NUMBER
C     LYING BETWEEN 0 AND 1
C     FROM GIVEN SEED IDUM


      FUNCTION RAN0(IDUM)
      INTEGER IDUM,IA,IM,IQ,IR,MASK
      REAL RAN0,AM
      PARAMETER (IA=16807,IM=2147483647,AM=1./IM,IQ=127773,IR=2836,
     *MASK=123459876)
      INTEGER K
      IDUM=IEOR(IDUM,MASK)
      K=IDUM/IQ
      IDUM=IA*(IDUM-K*IQ)-IR*K
      IF (IDUM.LT.0) IDUM=IDUM+IM
      RAN0=AM*IDUM
      IDUM=IEOR(IDUM,MASK)
      RETURN
      END
```